BEEBUG

FOR THE BBC MICRO & MASTER SERIES

What is the purpose of your visit?

What kind of shopping?

What is each age group likely to spend today?

Census

- MORE HIDE AND SEEK
- FASTER GRAPHICS
- POOL OF PERMS
- WORD SQUARE SOLVER

v12 n3

BEEBUG Vol.12 No.3 July 1993

FEATURES

Census 5
Word Square Solver 10
BEEBUG Education 12
Enhancing Basic's LISTO Command 15
A Pool of Perms 19
Public Domain Software 20
Gravity and Orbits (5) 22
BEEBUG Workshop: Faster Graphics 25
512 Forum 30
First Course: Sound (3) 34
More Hide and Seek 37
Very Big Numbers (2) 41
Mr Toad's Machine Code Corner 47

REGULAR ITEMS

Editor's Jottings/News 4
RISC User 50
Hints and Tips 51
Personal Ads 52
Postbag 53
Subscriptions & Back Issues 54
Magazine Disc 55

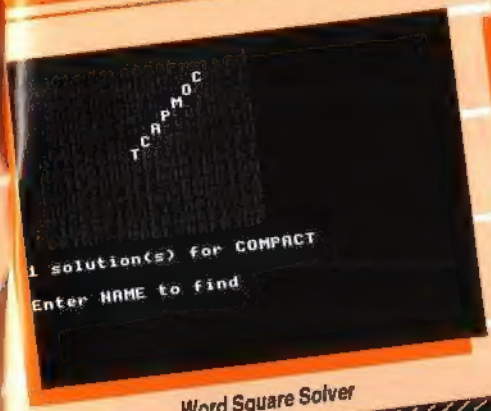
HINTS & TIPS

Changing Colours in Mode 7
Testing OSBYTE Routines
Wordwise Plus and VDU Codes

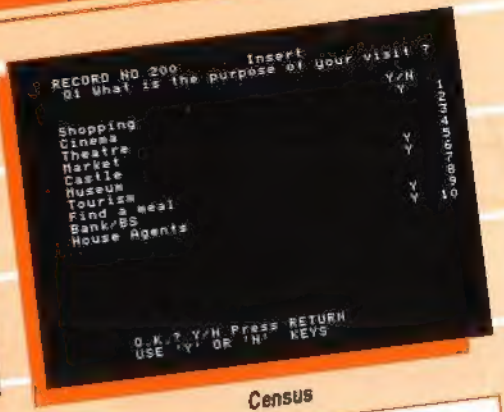
PROGRAM INFORMATION

All listings published in BEEBUG magazine are produced directly from working programs. They are formatted using LISTO 1 and WIDTH 40. The space following the line number is to aid readability only, and may be omitted when the program is typed in. However, the rest of each line should be entered exactly as printed, and checked carefully. When entering a listing, pay special attention to the difference between the digit one and a lower case l (L). Also note that the vertical bar character (Shift \) is reproduced in listings as |.

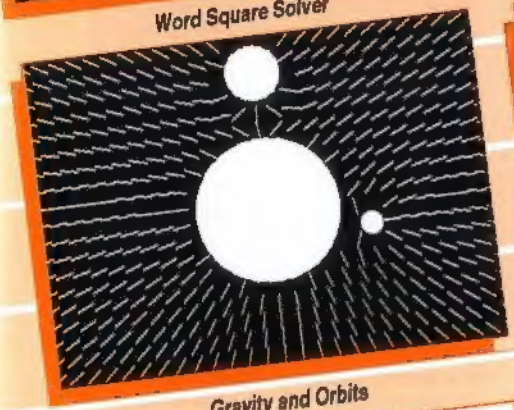
All programs in BEEBUG magazine will run on any BBC micro with Basic II or later, unless otherwise indicated. Members with Basic I are referred to the article on page 44 of BEEBUG Vol.7 No.2 (reprints



Word Square Solver



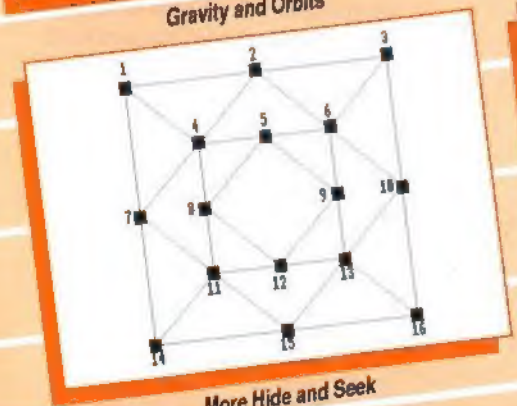
Census



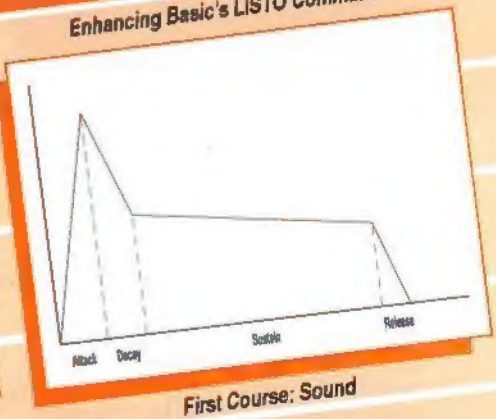
Gravity and Orbits

Bit	Option selected when bit set
0	Insert spaces after the line number
1	Insert spaces during FOR-NEXT
2	Insert spaces during REPEAT-UNTIL
3	Paging Mode active
4	Line splitting active
5	Printer active
6	Automatic mode selection active

Enhancing Basic's LISTO Command



More Hide and Seek



First Course: Sound

available on receipt of an A5 SAE), and are strongly advised to upgrade to Basic II. Any second processor fitted to the computer should be turned off before the programs are run.

Where a program requires a certain configuration, this is indicated by symbols at the beginning of the article (as shown opposite). Any other requirements are referred to explicitly in the text of the article.

Program needs at least one bank of sideways RAM.

Program is for Master 128 and Compact only.

Editor's Jottings/News

We have devoted all of this month's Postbag page to a letter from a reader which no doubt sums up the feelings of a good many subscribers following my editorial in Vol.12 No.1 about the likelihood that BEEBUG will cease publication in April 1994. Further feedback and suggestions from readers will be most welcome, and we will no doubt return to the topic again in the future.

However, what I really want to discuss is the question of upgrading to an Archimedes system. I know full well that many BEEBUG readers are quite satisfied and content with their existing system and see no reason to change, though they should bear in mind the future viability of a machine which will become more difficult to maintain as time goes by.

If you haven't experienced the speed and memory capacity of an Archimedes, you might be quite impressed at what it has to offer. Many programs written for a BBC micro will run directly on an Archimedes, and often much more effectively as we know ourselves. Graphic animations are a prime example. This ability to run BBC micro programs applies to many of the programs published in BEEBUG.

If a program will not run directly, then help is still at hand: Acorn supplies software with every Archimedes which enables it to emulate a BBC micro. This means that even machine code programs can be made to run, and it is perfectly feasible to use old favourites like View and Wordwise on an Archimedes.

I know from my own experience that while many users of the BBC micro initially view the different operating methods of the Archimedes with some disquiet, most of those who have upgraded rapidly become converted to the style of the Archimedes Desktop, and the use of the mouse to control much of what they do.

As readers will have seen from the leaflet enclosed with last month's issue of BEEBUG, we have put together a number of special offers on upgrading to an Archimedes system, and those offers are only available to BEEBUG readers. I would seriously urge readers to consider the pros and cons of upgrading, as these offers may not be available in the future.

The whole question of upgrading, and of software and hardware compatibility, was covered at some length in a two-part article in BEEBUG Vol.8 Nos.9 & 10. Much of what was said then still remains relevant, though the specific Archimedes models referred to may well have been superceded by later ones.

If you have any questions or queries relating to upgrade paths, then please let us know. We will try to answer such letters individually, but they may also provide us with the basis for further articles on the same subject. If you want to see more of the Archimedes range then why not visit our St Albans showroom?

ALL FORMATS COMPUTER FAIRS

We give below the dates of forthcoming *All Formats Computer Fairs* for reference.

25 Jul	National Motorcycle Museum, West Midlands (NEC/M42 J6)
11 Sept	National Motorcycle Museum, West Midlands (NEC/M42 J6).
12 Sept	Corn Exchange, Church Street, Brighton.
18 Sept	De Montford Hall, Granville St., Leicester.

NEXT ISSUE

Please note that the next issue of BEEBUG (for August/September) is due out mid August.

M.W.

Census (Part 1)

Paul Goldsmith helps you poll your own.

A social scientist friend asked me if my computer would analyse a questionnaire she wanted to use. I had not seen any software published for this purpose and I feel that there must be some demand for such, particularly for study purposes.

Sample input - question 1

The form of survey the program needed to deal with was such that a question has one or more pre-determined answers which can be selected or not selected (usually 'yes' or 'no'). To maximize the number of records and questions a compact method of recording data had to be found that could cope with multiple answers. This problem was solved by using a series of bits, which can be set or not set, represented as an integer. The answers to the questions are set in reverse order by the formula in line 2220 (and elsewhere) of *Census* i.e Number=2^(Total Number of Answers - Answer Set). Thus a positive answer number 3 of a 5 answer question gives a value of 4 (N%=2^(5-3)) and a positive answer number 2 of a 5 answer question gives a value of 8 (N%=2^(5-2)). The two together make 12 and this is what is stored so each record consists of one number per question held as integers.

TRY IT YOURSELF

Type in both the *Create* and *Census* programs as listed below. Line numbers must be strictly followed. At the end of both programs add the procedures in the 'Common' procedures listing - some judicious SPOOLing and EXECing will save time here. Save as *Create* and *Census* respectively.

Sample input - question 2

USING CREATE

In order to use any of the Census suite of programs it is necessary to type in text which reproduces the questions and answers on the questionnaire form. The form of this text must follow the conventions listed below:

CODE	A four Character Code
QUE	A suffix (obligatory)
Title	The survey title
Number Q	The number of questions
Number A	The maximum number of answers (up to 17)
Number R	The maximum number of records.
Number a1	The number of answers to question 1
Text t1	The text of question 1
Texts1-a1	The texts of the answers to question 1

The last 3 are repeated for each of the rest of the questions. For example, for a shopping survey the first few lines might be:

```
SHOP
QUE
Shopping Survey
8
17
200
4
What is the purpose of your visit?
Shopping
Cinema
Theatre
No response
3
What kind of shopping?
Groceries
Clothing  Other
```

And so on for the remainder of the 8 questions. The items must be consecutive with a Return at the end of each entry, left justified with no more than 18 characters per answer. They must be in straight ASCII with no formatting. This text, I call it the QUE text, must be saved as <code>qtx where <code> is the chosen four character code.

Run Create, select option one, 'Create files', and enter the name of your qtx file. This will create files CODE, <code>PAR and <code>QUE. When asked to create a <code>DA file answer 'Y'. When you quit the program you will get a 'Not Found' error - ignore this for the time being. If you have the magazine disc you will find SHOPqtx on it to try this out.

USING CENSUS

After using Create to create the necessary files you can run Census and enter the results of the survey. Records may be added in any order, within the limits of the maximum number set in Create.

Only the 'Insert' option will operate this month and result will not be saved, but this will give you chance to try things out and see how surveys are constructed via Create. It will enable a complete record to be entered starting with question 1. The screen invites the inputting of the number of a single answer, type the number of the answer and Return - a 'Y' will be placed next to that answer.

If a multiple answer has been given press Return at the 'Type number and Return' prompt. The cursor will go down each question in turn; press 'Y' for positive answers, any other key for negative answers. The 'Y's are displayed opposite the respective questions, 'Nos' are left blank. If the entry is incorrect you will be given the option to change it when you get to the end of the page. When all questions have been entered the record is saved and the menu reappears.

Always 'Quit' Census from the menu using option 7. This updates the file <code>PAR. Again you will get a 'Not Found' error on quitting, ignore this.

AND NEXT MONTH

Next month I will complete these programs and start to look at some of the analysis you can do.

Create

```
 10 REM Program Create
 20 REM Version B 1.2
 30 REM Author  Paul Goldsmith
 40 REM BEEBUG  July 1993
 50 REM Program Subject to Copyright
 60 :
100 MODE135:REM MODE 7 for BBC B
110 VDU26: CLS:CLEAR
120 PROCt(10,"C r e a t e")
130 REM menu
140 PROCwindow1
150 PRINT''"Create files"TAB(33)"1"
170 PRINT''"Quit         "TAB(33)"3"
180 VDU26:PROCwindow3
190 PRINT"Press Number":VDU26
200 A=GET
210 IF A=49 PROCreadtxt1:PROCcode:DIM
```

```
 T%(J%),N%(J%),A%(J%),A$(J%,H%):PROCread
txt2:PROCquest:VDU7:PROCmessage
  220 IF A=49 IF suff$="QUE" PROCdata
  230 IF A=50 PROCinst
  240 IF A=51 VDU26:CLS:CHAIN"SELECT"
  250 RUN
  260 :
 1000 DEF PROCdata
 1010 D$=CODE$+"DA":TIME=0:REPEAT UNTIL
TIME=500:CLS:PRINT"Create new ";D$;" Fil
e ";:INPUT A$
 1020 IF INSTR("Yy",A$) PROCda
 1030 ENDPROC
 1040 :
 1050 DEF PROCda
 1060 CLS:PRINT"Creating a new ";D$;" fi
le will"''"overwrite any existing ";D$+"
file":INPUT"Proceed Y/N "A$:IF INSTR("Yy
",A$) VDU7:C%=OPENOUTD$:EXT#C%=Rmax%*J%*
5:CLOSE#C%:PROCcode ELSE ENDPROC
 1070 ENDPROC
 1080 END
 1090 :
 1100 DEF PROCinst
 1200 ENDPROC
 1210 :
 1220 DEF PROCmessage
 1230 CLS
 1240 PROCwindow2
 1250 PRINT"File ";G$;" Created"
 1260 PRINT"file ";CODE$+"HED";" is also
"
 1270 PRINT"Required "
 1280 ENDPROC
 1290 :
 1300 DEF PROCreadtxt1
 1310 PROCt(8,"Read text")
 1320 PROCwindow2:INPUT"Name of text fil
e ";T$
 1330 ONERROROFF:T%=OPENINT$:PROCefile(T
%,T$)
 1340 PROCstring:CODE$=str$
 1350 PROCstring:suff$=str$
 1360 PROCstring:Head$=str$
 1370 PROCstring:J%=VAL(str$)
 1380 PROCstring:H%=VAL(str$)
 1390 PROCstring:Rmax%=VAL(str$)
 1400 ENDPROC
 1410 :
 1420 DEF PROCreadtxt2
 1430 FORY%=1 TO J%
 1440 PROCstring:A%(Y%)=VAL(str$)
 1450 FOR X%=0 TO A%(Y%)
 1460 PROCstring:A$(Y%,X%)=str$
 1470 NEXT:NEXT
 1480 CLOSE#T%
 1490 ENDPROC
 1500 :
 1510 DEF PROCstring
 1520 str$=""
 1530 REPEAT
 1540 N%=BGET#T%
 1550 IF N%=13 GOTO 1570
 1560 str$=str$+CHR$(N%)
 1570 UNTIL N%=13
 1580 ENDPROC
 1590 :
 1720 DEF PROCquest
 1730 G$=CODE$+suff$:B=OPENOUTG$
 1740 PRINT#B,Head$,J%:FOR X%=1 TO J%:PR
INT#B,A%(X%):FORY%=0 TO A%(X%):PRINT#B,A
$(X%,Y%):NEXT:NEXT
 1750 CLOSE#B:ENDPROC
 1760 :
 1770 DEF PROCcode
 1780 D%=OPENOUT"CODE":PRINT#D%,CODE$,J%
,H%,Rmax%:CLOSE#D%
 1790  P$=CODE$+"PAR":D%=OPENOUTP$:N%=0:
PRINT#D%,N%,N%,N%,N%,Rmax%:CLOSE#D%
 1800 ENDPROC
 1810 :
 1820 DEF PROCefile(i%,I$)
 1830 IF i%=0 PRINTTAB(3,19)"File "+I$+"
 not found":CLOSE#i%:TIME=0:REPEATUNTIL
TIME=200:CHAIN"CREATE"
 1840 ENDPROC
```

Census

```
   10 REM Program Census
   20 REM Version B 4.2
   30 REM Author  Paul Goldsmith
   40 REM BEEBUG  July 1993
   50 REM Program Subject to Copyright
   60 :
  100 ONERRORPROCerr
  110 CLEAR:PROCcode
  120 MODE135:DIM N$(H%),T%(H%),N%(H%),A
%(J%),A$(J%,H%):VDU 15:Rep$="":Rep1$="":
Miss$="":K%=17:REM MODE 7 for BBC B
  130 Start%=0:Numb%=0:End%=0:Top%=0
  140 PROCgetparam:PROCquest
```

```
  150 REM ******* M E N U *******
  160 INSERT=FALSE
  170 VDU26,15:CLS:PROCt(12,"M E N U")
  190 PRINTTAB(5,6)"INSERT          2"
  240 PRINTTAB(5,16)"QUIT           7"
  250 PRINTTAB(2,19);:VDU129:PRINT;Rep$:
Rep$="":PRINTTAB(23,4);:VDU130:PRINT"COD
E ";Code$
  260 PRINTTAB(29,19);:VDU131:PRINT"Star
t  ";Start%:PRINTTAB(29,20);:VDU131:PRIN
T"End    ";End%:PRINTTAB(29,21);:VDU131:
PRINT"Number ";Numb%
  270 *FX 15,1
  280 PRINTTAB(4,21);:VDU131:PRINT"PRESS
 NUMBER ":A=GET:IF A<49 OR A>55 VDU7:GOT
O280
  290 IN=FALSE:AM=FALSE
  310 IF A=50 PROCinsert
  360 IF A=55 PROCsaveparam:CHAIN"SELECT
"
  370 GOTO 150
  380 END
  390 :
 1000 DEF PROCenquire:HD$="Enquire":*FX2
00,0
 1060 ENDPROC
 1070 :
 1080 DEF PROCe
 1090 PROCscreen:N%=N%(T%):PROCdisplay:P
ROCins:PROCchange:PROCe
 1100 ENDPROC
 1110 :
 1120 DEF PROCchange
 1240 ENDPROC
 1250 :
 1260 DEF PROCinst
 1360 ENDPROC
 1370 :
 1380 DEF PROCinsert
 1390 HD$="Insert"
 1400 CLS:INPUTTAB(0,10)"WHICH RECORD ";
R%:IF R%<1 R%=1
 1410 IF R%>Rmax% GOTO 1400
 1420 C%=OPENUPD$:PROCefile(C%,D$)
 1430 PTR#C%=5*(R%-1)*Q%
 1440 B%=BGET#C%:PTR#C%=PTR#C%-1:IF B%=&
40 Rep$="Record "+STR$(R%)+" already exi
sts":VDU 7:CLOSE#C%:ENDPROC ELSE CLOSE#C
%
 1450 FOR T%=1 TO Q%
 1460 CLS:PROCscreen
 1470 PROCsingle:*FX202,32
 1480 PRINTTAB(5,22);SPC(25):INPUTTAB(5,
22)"O.K.? Y/N Press RETURN "Q$:IF LEN Q$
>1 GOTO1480
 1490 IF Q$<>"Y" GOTO 1460
 1500 IF Q$="Y" N%(T%)=N%:NEXT:PROCsaver
ec:Numb%=Numb%+1:PROCparam:Rep$="Record
No "+STR$(R%)+" Inserted"
 1510 ENDPROC
 1520 :
 1530 DEF PROCins
 1550 ENDPROC
 1560 :
 1570 DEF PROCdisplay
 1580 N1%=2^A%(T%)/2
 1590 FOR X%=1TOA%(T%)
 1600 IF N%>=N1% PRINTTAB(35,X%+3)"Y":N%
=N%-N1%:ELSE PRINTTAB(35,X%+3)" ":GOTO 1
610
 1610 N1%=N1%/2
 1620 NEXT
 1630 ENDPROC
 1640 :
 1650 DEF PROCscreen
 1660 CLS:PRINTTAB(1,1)"Q";T%;" ";A$(T%,
0);TAB(34,3)"Y/N"
 1670 FOR X%=1TOA%(T%)
 1680 PRINTTAB(0,3+X%)A$(T%,X%)TAB(30,3+
X%)X%
 1690 NEXT
 1700 PRINTTAB(0,0)"RECORD NO ";R%TAB(20
)HD$
 1710 ENDPROC
 1720 :
 1730 DEF PROCsaverec
 1840 ENDPROC
 1850 :
 1860 DEF PROCgetrec
 1870 *FX200,1
 1880 LOCAL T%
 1890 C%=OPENIND$:PROCefile(C%,D$)
 1900 N%=0:PORT%=1TOH%:N%(T%)=0:NEXT
 1910 IF 5*R%*Q%>EXT#C%:Rep$="Beyond end
of file":CLOSE#C%:ENDPROC
 1920 PTR#C%=5*(R%-1)*(Q%)
 1930 FOR X%=0 TO Q%-1
 1940 N%=BGET#C%:PTR#C%=PTR#C%-1:IF N%<>
&40 Rep$="Record "+STR$(R%)+" does not e
xist":CLOSE#C%:ENDPROC
 1950 INPUT#C%,I%:N%(X%+1)=I%
 1960 NEXT
```

```
 1970 CLOSE#C%
 1980 *FX200,0
 1990 ENDPROC
 2000 :
 2010 DEF PROCgetparam
 2020 C%=OPENINP$:PROCefile(C%,P$)
 2030 INPUT#C%,Start%:INPUT#C%,End%:INPU
T#C%,Numb%:INPUT#C%,Top%:INPUT#C%,Rmax%
 2040 CLOSE#C%
 2050 ENDPROC
 2060 :
 2070 DEF PROCcode
 2080 C%=OPENIN"CODE":PROCefile(C%,"CODE
")
 2090 INPUT#C%,F$,J%,H%,Rmax%
 2100 D$=F$+"DA":H$=F$+"HED":G$=F$+"QUE"
:Code$=F$+"******":P$=F$+"PAR":CLOSE#C%
 2110 ENDPROC
 2120 :
 2130 DEF PROCquest
 2140 B=OPENIN G$:PROCefile(B,G$)
 2150 INPUT#B,Head$,Q%:FOR X%=1 TO Q%:IN
PUT#B,T%:A%(X%)=T%:FORY%=0 TO A%(X%):INP
UT#B,A$(X%,Y%):NEXT:NEXT
 2160 CLOSE#B:ENDPROC
 2170 :
 2180 DEF PROCsingle
 2190 PRINTTAB(5,22);SPC(30)
 2200 INPUTTAB(5,23)"Type number and RET
URN "N$:IF VALN$=0 VDU7:PROCmulti:ENDPRO
C
 2210 N1%=VALN$:IF N1%>A%(T%) VDU7:GOTO
2200
 2220 N%=2^(A%(T%)-N1%):NR%=N%:PROCdispl
ay:N%=NR%:N%(T%)=N%
 2230 ENDPROC
 2240 :
 2250 DEF PROCmulti
 2260 PRINTTAB(5,23);SPC(30);TAB(5,23)"U
SE 'Y' OR 'N'  KEYS"
 2270 N%=0
 2280 FORX%=1TOA%(T%)
 2290 PRINTTAB(35,X%+3);:A=GET
 2300 IF A=89 N$(X%)="1":PRINT"Y":N%=N%+
2^(A%(T%)-X%)
 2310 IF A=78 N$(X%)="0":PRINT" "
 2320 N%(T%)=N%
 2330 NEXT
 2340 ENDPROC
 2350 :
 2360 DEF PROCchoose
 2370 CLS:INPUTTAB(10,10)"Which Question
 ";T%
 2380 ENDPROC
 2390 :
 2400 DEF PROCamend
 2470 ENDPROC
 2480 :
 2490 DEF PROCa
 2500 PROCscreen:N%=N%(T%):PROCdisplay:P
ROCins:PROCchange:N%=N%(T%):IF A=65 AND
AM=TRUE PROCsaverec:PROCa
 2510 ENDPROC
 2520 :
 2530 DEF PROCmove
 2620 ENDPROC
 2630 :
 2640 DEF PROCdelete
 2700 ENDPROC
 2710 :
 2720 DEF PROCshift
 2730 PROCselrec:R%=RF%:PROCgetrec
 2740 IF N%=0 Rep$="Record "+STR$(R%)+"
does not exist":ENDPROC
 2750 R%=RS%:PROCsaverec:Numb%=Numb%+1:I
F Rep1$<>"" Rep$=Rep1$:VDU7:ENDPROC
 2760 R%=RF%:Miss$="":PROCdelete
 2770 ENDPROC
 2780 :
 2790 DEF PROCselrec
 2800 PRINTTAB(3,17)"MOVE RECORD NO ";TA
B(20)"TO ":INPUTTAB(18,17);RF%:INPUTTAB(
25,17);RS%:IF RF%=0 OR RS%=0 OR RF%>Rmax
% OR RS%>Rmax% GOTO 2800
 2810 Skip=FALSE
 2820 ENDPROC
 2830 :
 2840 DEF PROCin
 2850 CLS:PROCscreen
 2860 PROCsingle
 2870 PRINTTAB(5,22);SPC(25):INPUTTAB(5,
22)"O.K.? Y/N Press RETURN "Q$:IF LEN Q$
>1 GOTO 2870
 2880 IF Q$="Y" N%(T%)=N%:PROCsaverec:Re
p$="Record No "+STR$(R%)+" Amended"
 2890 ENDPROC
 2900 :
 2910 DEF PROCparam
 2920 IF R%<Start% Start%=R%
 2930 IF R%>End% End%=R%
 2940 ENDPROC
```

continued on page 18

Word Square Solver

Kate Crennell helps solve your remaining word squares.

The BEEBUG Tenth Anniversary Competition (BEEBUG Vol.11 No.3 page 61) prompted me to look for an old program I wrote to solve any word square. I have modified it to work with that specific puzzle. It can also work alongside the word search generator first published in BEEBUG Vol.11 No.7, and updated in BEEBUG Vol.11 No.10.

Finding 'Compact'

Type in the listing and save it as WordSq2. Typing CHAIN"WORDSQ2" will show you a mode 1 screen and a prompt asking for the next name to look for. Type the letters of the word in upper case, because I have entered the word square in upper case. If the word is found, the whole square will be printed out in red, with the word highlighted in white. The program then continues to look for another occurrence of the word. The number of occurrences found is printed after the last one found. Words with few letters may occur several times. For example the words DFS, RAM, ROM all occur several times. To complete the puzzle, enter the words one at a time, and draw the word it finds on your copy. Note, the extra challenge word was 'COMPACT'. To stop the program press the Return key without entering any letters.

HOW THE WORD SQUARE IS STORED

The number of rows, *NR%*, and columns, *NC%*, of the square are read from the DATA statement in line 1420. They are used to dimension the array which holds the letters of the square. To use data for a different puzzle, change the DATA statement to fit the new puzzle, and then edit the lines 1440 to 1590 which hold the word square. There is no need for the puzzle to be a square, *NR%* and *NC%* are separate variables. I always use upper case letters, but you can use lower case if you prefer. The program searches for strings which can be any characters, numbers, letters or punctuation marks.

Finding 'ROM'

HOW IT WORKS

The main program sets up the data for the word square and then asks for the word to look for. *PROCFIND* looks for the word, and *PROCSHOW* prints it out. *PROCFIND* begins by storing the first and second letters of the word in *F$* and *S$*. Then it searches along the rows looking for *F$*. When *F$* is found, the row and column numbers are stored, and the procedure then looks all around the *F$* for *S$*; if it is found, the row and column numbers are stored. The program then looks for the rest of the

word using the positions of *F$* and *S$* to decide the direction in which to search.

In my first version of the program *PROCSHOW* made up a line of the square at a time, inserting the necessary colour changes. This is rather cumbersome, but has the advantage that it would work equally well on a printer with paper which moves only in one direction. In the version given here, the whole square is printed in red, and then the word found overprinted in white using TAB. This both runs more quickly and is easier to understand

```
   10 REM Program WordSq2
   20 REM Version B 1.0
   30 REM Author  K.M.Crennell
   40 REM BBEBUG  June 1993
   50 REM Program Subject to Copyright
   60 :
  100 REM NC%=NO OF COLUMNS IN PUZZLE, N
R%=NO OF ROWS.
  110 READ NC%,NR%
  120 DIM A$(NR%)
  130 FOR I%=1TONR%:READ A$(I%):NEXT
  140 MODE 1
  150 REPEAT.INPUT'"Enter NAME to find "
B$
  160 IF B$<>""PROCfind
  170 UNTIL B$=""
  180 END
  190 :
 1000 DEF PROCfind·REM find B$ in Wordsq
uare A$
 1010 N%=0:L%=LEN.B$)·IF L%>NC% AND L%>
NR% ENDPROC
 1020 F$=LEFT$(B$,1):S$=MID$(B$,2,1)
 1030 REM find first letter (F$)
 1040 FOR I%=1TONR%:J%=0
 1050 REPEAT:N%=INSTR(MID$(A$(I%),J%+1),
F$
 1060 IFN%>0 J%-J%+N%.PROCsecond
 1070 UNTILN%-0
 1080 NEXT:PRINT,NS%" solution(s) for ";
B$
 1090 ENDPROC
 1100 .
 1110 DEF PROCsecond·REM find second let
ter (S$) around first letter F$
 1120 A%-I%-1:B%-I%+1:IFA%<1 A%=1 ELSEIF
B%>NR% B%-NR%
 1130 C%-J% 1:D% J%+1:IFC%<I C%-1 ELSEIF
D%>NC% D%-NC%
 1140 FOR K% A%TOB%
 1150 FOR M%=C%TOD%
 1160 IF K%-I% AND M%-J% ELSEIF MID$(A$(
K%),M%,1)-S$ PROCrest
 1170 NEXTM%,K%
 1180 ENDPROC
 1190 :
 1200 DEF PROCrest:REM check rest of B$
 1210 X%-K%-I%.Y%-M% J%:U%-K%:V%-M%
 1220 IF L%-2 PROCshow:ENDPROC
 1230 FOR P%=3TO L%
 1240 U%-U%+X%:V%-V%+Y%
 1250 IF U%<1 OR U%>NR% OR V%<1 OR V%>NC
% ENDPROC
 1260 IF MID$(A$(U%),V%,1) <> MID$(B$,P%
,1) ENDPROC
 1270 NEXT:PROCshow
 1280 ENDPROC
 1290 :
 1300 DEF PROCshow:REM print solution
 1310 IFNS%>0 PRINT"More solutions, pres
s key to continue":G%=GET
 1320 CLS.PRINT:NS%=NS%+1:COLOUR1:REM pr
int square in red
 1330 FORP%=1TONR%.PRINT" "A$(P%):NEXT
 1340 U%=I%:V%=J%:COLOUR3:REM overprint
found word in white
 1350 FOR P%=1TOL%:PRINTTAB(V%,U%)MID$(B
$,P%,1)
 1360 U%=U%+X%:V%=V%+Y%
 1370 NEXT:PRINTTAB(0,NR%+1)
 1380 ENDPROC
 1390 :
 1400 REM dimensions of word square
 1410 REM number of columns, number of r
ows
 1420 DATA20,16
 1430 REM This is the word square follow
ing
 1440 DATASPITDNASTNIHDFWBLBTN
 1450 DATAQEAFLCTJLAASSEMBLERO
 1460 DATAPDCMAGSCANWSMMOCMFEE
 1470 DATADUCOUEAAPOBOCCOMAAEC
 1480 DATAFCRONIGUNMANRMEIRRSC
 1490 DATAIANAADFDJNSNPLOCGREH
 1500 DATARTREMWPPIOIAPYDROSTG
 1510 DATASIEAEOIRNRCVXUAORNJN
 1520 DATATOTTCRRAOTBEEPATPKEI
 1530 DATACNSENDIGGCUMARTBDCRT
 1540 DATAOFATEWGUBEEEARSDAIGN
 1550 DATAUGMARIKFILESECAABTOI
 1560 DATARAHDESOVIEWSSYURSPP
 1570 DATASMISFERTGABTSOPQMYDP
 1580 DATAESIDESEINTERWORDIOAR
 1590 DATACDRTROPEUGOLANAFRJBE
```

BEEBUG Education

by Mark Sealey

The day to day activities of the BBC micros rely on a special brand of technical competence which BEEBUG magazine has sustained and sponsored for over ten years.

It is somehow gratifying to note that this same world has also repeatedly attracted the attention of the academics - decidedly not from this 'hobbyist' point of view but because of one specific set of uses to which it has been put: in schools and colleges. The views of these 'experts' - as BEEBUG Magazine itself prepares to wind down and indeed the last 8-bit machine has ceased to be manufactured by Acorn - make a very important contribution to any assessment of the future of computers in schools.

So this month and in October BEEBUG Education looks at two recently published reports on the many uses for microcomputers in education. One reads them with mixed feelings. Knowing the strengths of our machines as we do, it is comforting that they are holding their place so well after this period of time

This ought to be due to a vigorous and original IT environment whose provision fully exploits the technical distinction of the platform and be evident everywhere throughout the education system. Patently, it is not Both reports make it clear that uptake and application still reflect only very dimly what is possible and - most seriously - what has been possible throughout almost the entire lifetime of BEEBUG.

In some ways we have not come as far as we should in that time; indeed the more theoretical of the two reports (reserved for examination next time) is in no doubt that IT (and hence the machines we use and which are covered by this column) "(has made)... a contribution to learning, but the contribution... (is)... not consistent across subjects or age bands"

The reports are published by the Department for Education and Kings' College London.

The more purely factual *Survey of Information Technology in Schools* summarises the results of a survey carried out over a year ago on a representative sample of nearly 900 primary and 550 secondary schools. It looks at hardware provision, access to and expenditure on equipment as well as the uses to which it is put, staff confidence, sources of information and advice. Finally the results of this survey are compared with those of previous years.

The most striking aspect - to BBC users - is how well our machine has held its share of the market: between them the model B and Master account for 53% of micros in primaries and 35% in secondaries. There is an estimated 132,500 micros in all schools compared with a third as many ten years ago.

The next biggest share in primaries (18%) goes to the Acorn 32-bit models; these are only 6% behind the second placed model in secondaries, too. Overall, nearly three quarters of equipment in primaries and over half in secondaries is still Acorn equipment. A healthy situation

MAINTENANCE AND FINANCING

The most common maintenance arrangement is still one where the LEA

(Local Education Authority) carries out the work and charges the school at cost, although a quarter of secondary schools have no formal arrangements; unless this conceals an imaginative resourcefulness on the part of technicians working late into the night, such a position is a little worrying!

Equally disturbing for anyone convinced of the benefits of IT in education is the abysmal sum spent on IT overall per pupil: little more than £13 (primaries) and £20 (secondaries) per annum... you cannot buy a subscription to BEEBUG for that

Surely we should value our children's access to computers more. More unsatisfactory still is the fact that, although nearly all pupils are shown in this survey as having regular access to micros, the average (inter-quartile) range of pupil to micro provision in primaries is at its best 18:1, usually closer to 25 or 30:1 and at its worst 115:1. In secondaries the ratio is never better than 12:1 Imagine how well you would have learnt if you had had to share your beloved machine with 19 other people!

Looking at funding a little more closely shows that in both primary and secondary schools, half of all IT is provided by the allowance paid directly to them by the LEA but that in primaries over a quarter of all equipment has come from voluntary sources (Parent Teacher Associations) etc. It is unlikely that this state of affairs will improve as LEAs' roles diminish and schools are increasingly underfunded as a result.

USES

Perhaps more important than any of this is how well micros are used to enable pupils to progress educationally. It is convenient here to make a distinction between primary and secondary schools.

In primaries the picture is as one would expect: all schools use (or claim to be using) their machines for English and maths, three quarters for science, art and technology, and very roughly half for history and geography.

Although the BBC micro is particularly well blessed with packages to handle musical output and has been from the very earliest days, fewer than 40% of primary schools are apparently using it to do so; perhaps this is because music is regarded as a specialist subject for infants and juniors with few teachers feeling confident enough to teach it.

Some 4% of primaries replying to the survey claimed to be using their computers in physical education (PE) It would be intriguing to know how; we could all learn something.

By far the most common use of computers in primaries is word processing. This is a little worrying, too, since separate evidence shows that many pupils are still encouraged to use the printing facilities of the computer as little more than a glorified typewriter. Think of the richness of software available to aid composition and trial and improvement - rather than fair copy - at the keyboard. You can only conclude that it is probably still the case that the facilities for redrafting of the system are still being woefully underused

The next most common application to which computers are put is games and puzzles. Again, think of the quality of some of these - and the plethora of unchallenging and ill thought-out dross that has existed from the beginning. The standard improved slightly as the developers came to know what wonders the 8-bit machines were capable of.

But at the same time, this survey also reveals that roughly a quarter of all hardware is over five years old. The same might well apply to software that at the time was doing its job adequately - particularly since over a third of all staff admitted to no other awareness of IT than through their initial training.

The other side of this somewhat tarnished coin is that well under an average of 10% of pupils regularly put their equipment to control or measurement uses - in either educational phase. Set this against the outstanding and pioneering facilities for interfacing with the real world supported by nearly half the micros in schools, 8-bit BBCs with a built in user port!

SECONDARIES

In secondaries, a pleasingly high proportion of staff are regularly using IT in support of most areas of the curriculum including special needs, classics, PE and administration; predictably, secondaries make much heavier use of on-line sources than primaries, Prestel and Campus 2000, NCET and the now shamefully defunct NERIS being the most used. Surprisingly, perhaps, only the staff from secondary business and computer studies departments rated themselves anywhere near as confident in using computers with children as the 72% of all primary teachers. Modesty, or genuine unpreparedness?

Data handling by secondary teachers is spread out healthily across the age range, and about a third of all secondary teachers claim to use a micro at least twice a week in their teaching - even a fifth of all drama, but only the same proportion of modern language teachers

Again, there are some surprises in the underuse suffered by pupils of secondary age: only an average of 4% of IT use was devoted to musical composition and not much more to simulations - surely two of the activities at which computers are best. The survey suggests that secondary age pupils spend about 8% of their lesson time using IT; it is hard to reconcile this with other statistics from the survey already commented upon

CONCLUSIONS

The DfE Survey draws few overall conclusions of its own; but among them is one that over six out of every ten teachers regularly use micros in their teaching; extrapolating carefully from this statistic, it can be very roughly estimated, then, that 100,000 teachers have used or are using a BBC 8-bit micro! A third (compared with a fifth five years previously) of primaries reported that IT has made a "substantial contribution to teaching and learning".

Clearly, the picture is uneven to say the least. Opportunities are being missed, and there are several facts that do not bode well for the future It has to be admitted that there is still no consensus on what sort of contribution IT can and should make to formal education, and how great that contribution is.

One recent attempt to define these, and what implications any such definition holds for future educational use of the Acorn family of micros will be examined in greater depth next time when attention turns to the Kings' College report.

Statistical Bulletin 6/93, Survey of Information Technology in Schools. February 1993, ISSN 0142-5013 - f.o.c.

Department for Education, Analytical Services Branch Room 338, Mowden Hall Staindrop Road, Darlington Co. Durham DL3 9BG tel: 0325 392683.

Enhancing Basic's LISTO command

Make program listings crystal clear with Stephen Ramplin's LISTO enhancement.

The layout of a program as listed can be controlled by the LISTO command Spaces may be inserted for the duration of all FOR-NEXT and REPEAT-UNTIL loops and immediately after the line number

The utility presented here enhances the LISTO feature built into Basic. It provides automatic selection of a specified screen mode, printer and paging modes and line splitting of multiple statement lines at the colons.

Enter the program as listed below After checking for errors, save it and then run it. The resulting code is saved to disc under the filename *Utility*. To use it in the future, enter *RUN Utility and it will be loaded and run

The various features provided are selected by setting the appropriate bits of the LISTO flag. The table below shows which options are activated when a particular bit is set

Bit	Option selected when bit set
0	Insert spaces after the line number
1	Insert spaces during FOR-NEXT
2	Insert spaces during REPEAT-UNTIL
3	Paging Mode active
4	Line splitting active
5	Printer active
6	Automatic mode selection active

You can use LISTO followed by a denary number; LISTO 49 would give you spaces after the line number, line splitting and the printer (1+16+32). For less mental arithmetic use the command LISTO % followed by a binary number, for example LISTO %11001. Here bits four, three and zero are set. Therefore, spaces would be inserted after the line numbers, paging mode would be on and line splitting would occur.

```
1000DEF PROCfind:REM find B$ in Wordsquare A$
1010NS%=0:L%=LEN(B$):IF L%>NC% AND L%>NR% ENDPROC
1020F$=LEFT$(B$,1):S$=MID$(B$,2,1)
1030REM find first letter (F$)
1040FOR I%=1TONR%:J%=0
1050REPEAT.N%=INSTR(MID$(A$(I%),J%+1),F$)
1060IFN%>0 J%=J%+N%:PROCsecond
1070UNTILN%=0
1080NEXT.PRINT;NS%" solution(s) for ";B$
1090ENDPROC
1100:
```

Standard listing

To specify which screen mode is used you can enter the command LISTO MODE followed by the mode number which must be in the range 0 to 7. TOP is checked by the utility to make sure that selecting that mode doesn't wipe off the end of your program. The default is mode 7. To switch off automatic mode selection clear bit 6 of the LISTO flag, e.g. LISTO 0.

Using this utility, you can have long program lines, saving memory, and still have clear readable listings. To use a cliche, you'll wonder how you ever managed without it!

HOW IT WORKS

The program implements the LISTO % and LISTO MODE commands by intercepting the BRK vector. A character-entering-buffer event is also enabled and the input buffer checked for the LIST command every time the return key is pressed

```
1000 DEF PROCfind
     REM find B$ in Wordsquare A$
1010 NS%=0
     L%=LEN(B$)
     IF L%>NC% AND L%>NR% ENDPROC
1020 F$=LEFT$(B$,1)
     S$=MID$(B$,2,1)
1030 REM find first letter (F$)
1040 FOR I%=1TONR%
     J%=0
1050   REPEAT
     N%=INSTR(MID$(A$(I%),J%+1),F$)
1060     IFN%>0 J%=J%+N%
     PROCsecond
1070   UNTILN%=0
1080 NEXT
     PRINT;NS%" solution(s) for ";B$'
1090 ENDPROC
1100
```

Listing after using enhanced LISTO

When the LIST command is detected the LISTO flag is examined. If the appropriate bits are set, the screen mode is changed, paging mode is activated and the printer is switched on. Next the write character vector is redirected allowing the utility to examine each character of a program as it is listed.

When a colon is found, and if bit four of the LISTO flag is set, then the line is split. The program also tests for quotes to prevent line splitting in situations like PRINT":". Finally the routine determines that a program has finished listing, or if listing was terminated by pressing Escape, when the '>' character is printed in command mode.

Once a program has been listed the printer and paging mode are turned off and the write character vector is reset. I hope you find this utility as useful as I have.

```
  10 REM Program Listo+
  20 REM Version B1.10
  30 REM Author  Stephen Ramplin
  40 REM BEEBUG  July 1993
  50 REM Program subject to copyright
  60 :
 100 MODE7
 110 PROCvariables
 120 PROCassemble
 130 PRINT'"Press Space to save code"
 140 REPEATUNTILGET=32
 150 OSCLI"SAVE Utility 900 "+STR$~P%
 160 PRINT'"To activate *RUN Utility"
 170 END
 180 :
1000 DEFPROCvariables
1010 brkv=&202:evntv=&220
1020 wrchv=&20E:ptr=&FD
1030 brkcpy=&70:evntcpy=&72
1040 wrchcpy=&74:top=&12
1050 mode=&76 listo=&77
1060 tempX=&78:tempY=&79
1070 quotes=&7A:input=&700
1080 listoB=&1F:ptrA=&0B
1090 osbyte=&FFF4
1100 oswrch=&FFEE
1110 ENDPROC
1120 :
1130 DEFPROCassemble
1140 FORpass%=0TO2STEP2
1150 P%=&900
1160 [OPT pass%
1170 .init
1180 LDAbrkv:STAbrkcpy
1190 LDAbrkv+1:STAbrkcpy+1
1200 LDA#edlisto MOD256
1210 LDX#edlisto DIV256
1220 SEI
1230 STAbrkv
1240 STXbrkv+1
1250 CLI
1260 LDAevntv:STAevntcpy
1270 LDAevntv+1:STAevntcpy+1
1280 LDA#cline MOD256
1290 LDX#cline DIV256
1300 SEI
1310 STAevntv
1320 STXevntv+1
1330 CLI
1340 LDA#0:STAlisto:STAquotes
1350 LDA#7:STAmode
1360 LDA#14:LDX#2
1370 JMPosbyte
```

```
1380 :
1390 .doError
1400 PLA:TAY:PLA:TAX:PLA
1410 JMP brkcpy)
1420 :
1430 .edlisto
1440 PHA:TXA:PHA:TYA:PHA
1450 LDY#0
1460 LDA(ptr),Y
1470 CMP#26:BNE doError
1480 :
1490 .NoSuchVar
1500 DEY:JSRspaces
1510 CMP#&C9:BNE doError
1520 INY:LDAinput,Y
1530 CMP#79:BNE doError
1540 JSRspaces
1550 CMP#ASC"%":BEQ CalcDec
1560 :
1570 .GetMode
1580 CMP#&EB:BNE doError
1590 JSRspaces
1600 CMP#48:BCC doError
1610 CMP#ASC"8":BCS doError
1620 AND#&07
1630 STAmode
1640 LDA#64
1650 ORAlistoB
1660 STAlistoB
1670 JMPfinish
1680 :
1690 .CalcDec
1700 LDA#0:STAlisto
1710 .cdloop
1720 JSRspaces
1730 CMP#&0D:BEQcdcopy
1740 SEC:SBC#ASC"1"
1750 ROLlisto
1760 JMPcdloop
1770 :
1780 .cdcopy
1790 LDAlisto:STAlistoB
1800 .finish
1810 BRK:BRK:EQUW&B0B:BRK
1820 :
1830 .spaces
1840 INY:LDAinput,Y
1850 CMP#ASC" ":BEQspaces
1860 RTS
1870 :
1880 .exit
1890 LDXtempX:LDYtempY
1900 .ignore
1910 LDA#2:JMP(evntcpy)
1920 :
1930 .cline
1940 CMP#2:BNEignore
1950 CPY#&0D:BNEignore
1960 LDAptrA+1
1970 CMP#7:BNEignore
1980 STXtempX:STYtempY
1990 LDX#0:LDY#&FF
2000 JSRspaces
2010 :
2020 .clloop
2030 CMPcommand,X:BEQnoabbrv
2040 CMP#ASC".":BNE exit
2050 CPX#0:BEQexit
2060 INY:LDAinput,Y
2070 JMP Check0
2080 :
2090 .noabbrv
2100 INY:LDAinput,Y
2110 INX:CPX#4:BNEclloop
2120 .Check0
2130 CMP#ASC"O":BEQexit
2140 :
2150 .testBit6
2160 LDAlistoB
2170 AND#&40:BEQtestBit3
2180 LDXmode:LDA#133
2190 JSRosbyte
2200 CPYtop+1:BCCtestBit3
2210 BEQtestBit3
2220 LDA#22:JSRoswrch
2230 LDAmode:JSRoswrch
2240 .
2250 .testBit3
2260 LDAlistoB
2270 AND#&08:BEQtestBit5
2280 LDA#14:JSRoswrch
2290 :
2300 .testBit5
2310 LDAlistoB
2320 AND#&20:BEQredirect
2330 LDA#2:JSRoswrch
2340 :
2350 .redirect
2360 LDAwrchv:STAwrchcpy
2370 LDAwrchv+1:STAwrchcpy+1
```

```
2380 LDA#listing MOD256
2390 LDX#listing DIV256
2400 STAwrchv.STXwrchv+1
2410 JMP exit
2420 :
2430 .listing
2440 PHA
2450 CMP#34:BNE tstArrow
2460 LDA#&FF
2470 EORquotes
2480 STAquotes
2490 JMPprintchr
2500 :
2510 .tstArrow
2520 CMP#ASC">":BNE tstColon
2530 LDAptrA+1
2540 CMP#7:BNEprintchr
2550 .reset
2560 LDAwrchcpy:STAwrchv
2570 LDAwrchcpy+1:STAwrchv+1
2580 LDA#3:JSRoswrch
2590 LDA#15:JSRoswrch
2600 PLA
2610 JMPoswrch
2620 :
2630 .tstColon
2640 CMP#ASC":":BNEprintchr
```

```
2650 :
2660 .testBit4
2670 LDAlistoB
2680 AND#&10.BEQprintchr
2690 LDAquotes:BMIprintchr
2700 TXA:PHA.TYA:PHA
2710 LDA#&0D:JSRvector
2720 LDA#&0A:JSRvector
2730 LDX#5.LDAlistoB
2740 AND#1:BEQnoinx
2750 INX
2760 .noinx
2770 LDA#ASC" "
2780 .slloop
2790 JSRvector
2800 DEX:BNEslloop
2810 .
2820 .restore
2830 PLA TAY PLA:TAX:PLA
2840 RTS
2850 .
2860 .printchr PLA
2870 .vector JMP(wrchcpy)
2880 .command EQUS"LIST"
2890 ]
2900 NEXTpass%
2910 ENDPROC
```

Common (continued from page 6)

```
T%,& Numb%:PRINT#C%,Top% PRINT#C%,Rmax%
2990 CLOSE#C%
3000 ENDPROC
3010 :
3020 DEF PROCparameter
3160 ENDPROC
3290 :
3300 DEF PROCerr
3310 REPORT:PRINT" at line "ERL
3320 PRINT"Press a key":A=GET
3330 ENDPROC
3340 :
3350 DEF PROCefile(i%,I$)
3360 IF i%=0 PRINTTAB(3,19)"File "+I$+"
 not found":CLOSE#i%:TIME=0:REPEATUNTIL
TIME=200:CHAIN"SELECT"
3370 ENDPROC
```

Common Procedures

```
4000 REM Common procedures
4010 :
4020 DEFPROCwindow1
4030 VDU28,3,20,36,5
4040 ENDPROC
4050 :
4060 DEFPROCwindow2
4070 VDU28,3,21,36,15
4080 ENDPROC
4090 :
4100 DEFPROCwindow3
4110 VDU28,3,24,36,22
4120 ENDPROC
4130 :
4140 DEFPROCblue
4150 FOR L1%=0 TO 25
4160 VDU148,157,135,13,10
4170 NEXT
4180 ENDPROC
4190 :
4200 DEFPROCt(T%,H$)
4210 PROCblue
4220 VDU30:FOR L1%=1 TO 2:VDU131,157,12
9,141.PRINTTAB(7)"C  E  N  S  U  S":NEXT
:FOR L%=1 TO 2:VDU148,157,129,141:PRINTT
AB(T%)H$:NEXT
4230 ENDPROC
```

A Pool of Perms

Geoffrey Dodds presents a utility for change ringers.

The letter from Mr R.J.Lindsell (BEEBUG Vol.11 No.10) seeking a way of generating permutations is of interest to me because, as a change ringer, I often require tables of permutations to assist in proving that no repetition of a 'row' or permutation occurs in a peal composition.

In English change-ringing, bells are numbered in sequence down a diatonic scale. At each rotation of the bells a symmetrical pattern of changes is woven which is repeated using linking changes, generating fresh rows with each change until the initial row is rung again. The main aim is to achieve this without any other row being repeated. By varying the linking changes, the composer of a peal (of over 5000 different rows) needs to generate only the first row in each pattern, by transposing rows by row operators, and checking that none of these are repeated. To help with this the simple program listed below generates permutations in numerical order.

The number to be permuted, *N%*, is input and used to DIM the array of numbers and a checking array, marking those numbers used in columns 1 to (*col%*-1), when finding a number to place in *perm%*(*col%*). This is generally quicker than loop testing each of the previous columns. To initiallise the procedure, *col%*=2 is set and *stop%* is made FALSE.

When all permutations have been found, 'Complete' is printed and the program stops. Of course, instead of merely printing the permutations, they can be stored or processed for use within a larger program.

The *PROCpermute* routine exits with *col%*=*N%*5+1. On re-entry, the *N%*th column is examined, its number is taken into *B%* and is also cancelled in array, *check%*. *B%* is then incremented to the next unused number, up to the limit, *N%*, placed in *perm%*(*col%*) and *check%*(*B%*) is set. However, if *B%*=*N%*, then the number in the previous column is dealt with, followed by each subsequent column until the permutation is complete (*col%*>*N%*). Finally, when the last permutation, *N%*,..3,2,1, has been found, *stop%* is set to TRUE and the program will end.

```
   10 REM Program Perms
   20 REM Version B 1.0
   30 REM Author  Geoffrey Dodds
   40 REM BEEBUG  July 1993
   50 REM Program subject to copyright
   60 :
  100 INPUT"(To pause press Space bar)"'
"Number to be permuted ",N%
  110 DIM perm%(N%),check%(N%,
  120 @%=3:col%=2:stop%=FALSE
  130 REPEAT PROCpermute
  140 IF stop%=TRUE UNTIL TRUE:PRINT'"Co
mplete":END
  150 FOR I%=1 TO N%:PRINT,perm%(I%);:NE
XT:PRINT
  160 IF INKEY(0)=32 Z$=GET$
  170 UNTIL FALSE
  180 :
 1000 DEF PROCpermute
 1010 LOCAL B%
 1020 REPEAT col%=col%-1:B%=perm%(col%):
check%(B%)=0
 1030 REPEAT IF B%=N% UNTIL TRUE.UNTIL c
ol%=1:stop%=TRUE:ENDPROC
 1040 B%=B%+1
 1050 IF check%(B%)=1 UNTIL FALSE
 1060 perm%(col%)=B%:check%(B%)=1:col%=c
ol%+1
 1070 IF col%<=N% B%=0:UNTIL FALSE
 1080 UNTIL TRUE:UNTIL TRUE
 1090 ENDPROC
```

PD Software

by Alan Blundell

Some advice for inexperienced BBC owners interested in the Public Domain and some details of new software are covered in this month's column by Alan Blundell.

You hear some odd things, if you run a PD library. Apart from copying lots of discs and amassing a wide range of software, there are the letters you receive. I've been thinking about this recently, partly because the number of letters which I receive seems to continue unabated and partly because of the apparent trend for newcomers who are still acquiring a BBC micro and getting to grips with it.

I would have thought at this stage in the life of the Beeb that the average user was someone whose micro fits like a well-worn glove, who was familiar with all the ins and outs of 8-bit computing and who had picked up all the understanding needed to use their micro for whatever purposes they own it. Apparently not; I regularly hear from people who have just acquired a BBC Micro and are looking for software to use with it. Take, for an (unusual) example, the user who had gone from a BBC micro many years ago, to an Atari ST, to an Amiga, to an IBM PS/2 and back recently to a BBC. It is (yet more) testimony to the design of the BBC micro that this is still happening.

When you are not familiar with a computer, you need support in order to become familiar with its ways, and many of these new users are looking for ways to get to grips with their machines. One way of doing that is by reading relevant magazines and picking things up along the way, but apart from BEEBUG, there is very little available in this line. When some people seem to be without even a manual with their micro, it must be difficult to get anywhere at all!

With these thoughts in mind, and with experience of some problems which people have had, I thought that I would briefly try to help if you are one of those who relies on a disc doing something when you press Shift-Break. If not, please bear with me; there is some new PD software news later on.

WHAT'S ON THE DISC?

Not all PD libraries and user groups add a menu system to their discs, and all have at least some discs without menus or for which menus are not appropriate. Examples include collections of clip art, text files, screen images and similar non-program software. To find out what is on a disc (DFS or ADFS), use the command '*CAT'; this can be abbreviated to just '*.' (all BBC OS commands can be abbreviated to some degree, so long as the command ends in a full stop and contains sufficient letters to identify it uniquely). After that, the way you deal with files varies; clip art, as a particular example, should be detailed in the manual for your DTP package (if you don't have a manual, are you sure that you have a legal copy of the program?).

Text files, probably the most common problem area, are quite easy to deal with, using the **TYPE* command; simply type **TYPE <filename>*, replacing *<filename>* with the actual name of a file, of course.

Use Shift and Ctrl together to pause the output if there is more than a screenful of text. Alternatively, if you have a Master series micro, you can load the file into the built-in text editor, EDIT, by typing **EDIT <filename>*, or if you have a word processor, load the file into that. Either of the latter two alternatives is preferable, simply because the *TYPE command does not let you scroll back if you have missed something. A text editor or word processor gives you more control.

Programs which are not accessed via a menu are usually Basic or machine code. There are straightforward ways of telling the difference, but if you have no experience, try *CHAIN"filename"* or **RUN filename*. CHAIN"" is used to run Basic programs; *RUN does the same for machine code programs. If neither of these works, assume the file isn't a program, at least until you have the experience to investigate further.

That's the end of my brief survival guide. It probably contains nothing at all new to many, but I know that some readers will find it useful as a starting point.

FORTH COMPILER

It may seem incongruous to go straight from a beginners introduction to the FORTH language, but Martin Rigby has given me permission to distribute as PD a FORTH language compiler. The compiler is based on a standard source code file released by the Forth Interest Group, for another computer (actually the Rockwell development machine *System 65*). Whilst the source is written using 6502 assembler mnemonics, there must still have been a significant amount of effort involved on Martin's part, since the OS calls which the program must use are completely different.

He has converted most of the relevant calls, but warns that the result is not yet a complete FIG-Forth compiler, notably in not yet having disc-related functions included. However, the full source code is included, and is heavily commented, so if you have a real interest, there is plenty of opportunity to get involved and develop the compiler further.

The source is compatible with "ASM" and "ADE" by System Software, and with Alan Phillips' excellent ROM-based public domain assembler, which has been mentioned previously in this column.

BIBLE SEARCH

Readers may remember a recent PD column in which I gave the story of how I converted the full text of the King James bible from an MS-DOS version. The story was accompanied by a request for a search program for use with the resulting 5Mb of text files. Paul Goldsmith recently sent me just such a program, which does the job admirably. Thanks to him for his efforts; copies of the program have already been passed to interested parties.

NEW AGE

Geoff Gibbs has prepared a comprehensive set of programs on astrology, which will prepare an astrological chart from information which you supply and a collection of files on the subject of 'New Age' philosophy. To quote, "With the world in such a mess and the shortcomings of science and the politicians, it is time that alternative ideas are thought about." If you want to know something about the subject, this is perhaps a good place to start.

That's it for this issue. Next time, I will treat you to an insight into the lighter side of running a PD library.

Gravity and Orbits (Part 5)

Cliff Blake rounds off his series on orbits.

Our last program does not involve the movement of a spaceship, but shows the technique applied to mapping the gravity field around a number of bodies. To reduce ratios, the mass of each body is assumed to be directly proportional to its radius.

THE PROGRAM

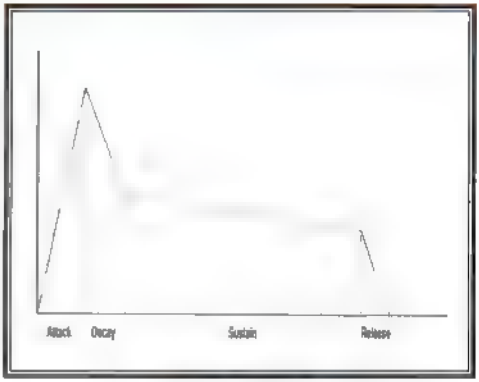
Up to seven bodies may be specified, and a choice of plotting zero slope contours, or maximum slope lines-of-force is given. The centre of each body is set by using the cross cursor and pressing key C, followed by moving the cursor to a point on the required circumference and pressing key R to determine the radius

As a first trial choose 2 bodies and zero slope contours. Position the cursor near to the lower left corner, and press 'C'. Then move the cursor to the right, halfway across the screen, and press 'R'. A disc will be drawn, which may be partially off screen. Move the cursor to the top right corner, and repeat, but make the radius only a couple of centimetres. After the last disc is completed, the screen clears before redrawing them and plotting sections of contour lines. These are similar to height contours on maps, but in this case represent levels in gravity pits.

To imitate the field around the nodal moon of part 4, plot six smaller discs in a circle, then cover them with a larger disc.

When reviewing lines-of-force, note that only an initially stationary object would start to track along a line. Moving objects crossing a line, will have their speed/direction modified by the force.

(R)epeat or (Q)uit can only be selected when mapping is finished



The first trial

YOUR OWN DEMONSTRATIONS

If you want to try your hand at simulated experiments, here are some suggestions.

a) Increment a swinging pendulum and show that its motion is sinusoidal.
b) Introduce air resistance and show that the swing dies away exponentially.
c) If you are into electrics, apply the same requirements to a tuned circuit. You'll need to use something like V=R*I, V=L*dI, I=C*dV, where dI and dV are the increments corresponding to mechanical velocity changes.
d) Show how the level in a draining water tank or discharging capacitor varies.
e) Demonstrate the final positions taken up by corks carrying repelling bar magnets, floating in water.

I have done (a), and a problem similar to (c), so they are feasible.

PROGRAMMING TECHNIQUES

From the start it is best not to get bogged down with Laws-of-Motion in actual units of distance, velocity etc. The position of the spaceship will be defined by graphic X,Y values, and velocity movement can be simulated by incrementing with further

appropriate X,Y step values. Thus: Xs=Xs+Xv. Similarly the velocities can be incremented to represent accelerations: Xv=Xv+Xgp+Xgm, Xv=Xv+Xt

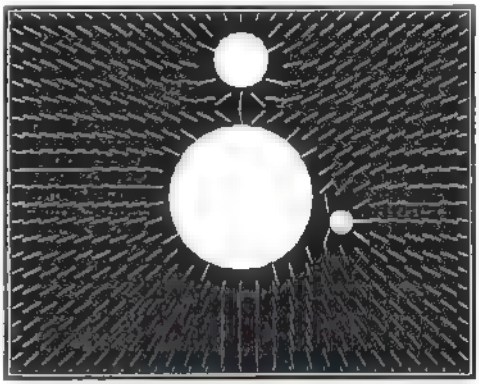
All the programs in this series follow the principle of calculating the next move at each incremental step. Where a variable is common to more than one program, it has the same name Similar variables have similar names to fit the application Thus: Xdp,Ydp for distances to planet, and Xdm,Ydm for distances to moon. The variables of the *LUNAR* program in part 2 form a useful example, and are:

	SPACESHIP-from-PLANET
Xdp,Ydp	Distances
Rdps	Radial distance squared
Rdp	Radial distance
	SPACESHIP-from-MOON
Xdm,Ydm	Distances
Rdms	Radial distance squared
Rdm	Radial distance
	SPACESHIP
Xs,Ys	Position
Xv,Yv	Positional steps due to velocities
Xgp,Ygp	Change in velocity steps due to planet's gravity
Xgm,Ygm	Change in velocity steps due to moon's gravity
Xt,Yt	Change in velocity steps due to propulsion thrust

The initial aim should be to get a satisfactory graphic display on the screen by allocating suitable values to variables and constants. If required, apparent actual units for display can be obtained by using empirical fiddle factors. For example vel%=INT(.85*Rv). Trigonometrical conversions should be avoided to allow the program to increment at an acceptable rate.

Having calculated a radius using the form R=SQR(X*X+Y*Y), it is unnecessary to derive the angle in order to obtain the trigonometrical components, because SIN(angle)=Y/R, COS(angle)=X/R, and TAN(angle)=Y/X. For instance, in Xgp=20000*Xdp/Rdp/Rdps:

a) Xdp/Rdp is COS(angle),
b) /Rdps is inverse square of distance,
c) 20000 is an arbitrary mass constant to give acceptable movement on screen



A more complex map

Sketching each basic vector diagram on paper is a great help in finding ways to simplify the calculations. In cases where an angle is required, consideration should be given as to whether it is necessary to turn it into degrees. Usually calculations can remain in radians, while degrees are used for display only.

I hope you come up with some interesting projects - have fun!

```
 10 REM Program MAP
 20 REM Version B2.0
 30 REM Author  Cliff Blake
 40 REM BEEBUG  July 1993
 50 REM Program subject to copyright
 60 :
100 DIM Xp(7),Yp(7),Rp(7)
110 REPEAT
120 MODE7
130 PROCinfo:g%=GET
140 MODE0
150 PROClayout:PROCplanets
```

```
 160 PROCauto:VDU7
 170 REPEAT.g$=GET$:UNTIL INSTR("RrQq",
g$)
 180 g%=ASC(g$)
 190 UNTIL g%=81 OR g%=113
 200 CLG:MODE7
 210 END
 220 :
1000 DEF PROCauto
1010 FOR y=1015 TO 7 STEP -48
1020 FOR x=15 TO 1263 STEP 48
1030 PROCcalc
1040 IF accept%=TRUE THEN PROCplot
1050 NEXT x
1060 NEXT y
1070 ENDPROC
1080 :
1090 DEF PROCplot
1100 IF c%=1 MOVE x+Ym,y-Xm:DRAW x-Ym,y
+Xm
1110 IF c%=2 MOVE x+Xm,y+Ym:DRAW x-Xm,y
-Ym
1120 ENDPROC
1130 :
1140 DEF PROCcalc
1150 Xgt=0:Ygt=0
1160 accept%=TRUE
1170 FOR n%=1 TO p%
1180 PROCdistance
1190 IF accept%=TRUE THEN PROCgravity
1200 NEXT n%
1210 ENDPROC
1220 :
1230 DEF PROCdistance
1240 Xd=Xp(n%)-x:Yd=Yp(n%)-y
1250 Rds=Xd*Xd+Yd*Yd:Rd=SQR(Rds)
1260 IF Rd<Rp(n%)+24 THEN accept%=FALSE
1270 ENDPROC
1280 :
1290 DEF PROCgravity
1300 Xg=Rp(n%)*Xd/Rd/Rds:Yg=Rp(n%)*Yd/R
d/Rds
1310 Xgt=Xgt+Xg:Ygt=Ygt+Yg
1320 Rgts=Xgt*Xgt+Ygt*Ygt:Rgt=SQR(Rgts)
1330 Xm=24*Xgt/Rgt:Ym=24*Ygt/Rgt
1340 ENDPROC
1350 :
1360 DEF PROClayout
1370 *FX4,1
1380 GCOL3,1
1390 FOR n%=1 TO p%
1400 x=640:y=512:PROCcursor
1410 VDU4:PRINT TAB(0,29)SPC(79)
1420 PRINT TAB(2,29)"Centre of Planet "
;n%;" Then press key C":VDU5
1430 REPEAT
1440 PROCmove
1450 UNTIL g%=67 OR g%=99
1460 Xp(n%)=x:Yp(n%)=y
1470 x=x+2:y=y-2:PROCcursor
1480 VDU7
1490 VDU4:PRINT TAB(0,29)SPC(79)
1500 PRINT TAB(42,29)"Radius of Planet
";n%;" Then press key R":VDU5
1510 REPEAT
1520 PROCmove
1530 UNTIL g%=82 OR g%=114
1540 Xr=Xp(n%)-x:Yr=Yp(n%)-y
1550 Rp(n%)=SQR(Xr*Xr+Yr*Yr)
1560 PROCcursor
1570 VDU7
1580 PROCdisc(Xp(n%),Yp(n%),Rp(n%))
1590 NEXT n%
1600 GCOL0,1
1610 *FX4,0
1620 ENDPROC
1630 :
1640 DEF PROCmove
1650 g%=GET
1660 PROCcursor
1670 IF INKEY(-1) THEN inc%=20 ELSE inc
%=4
1680 IF g%=136 THEN x=x-inc%
1690 IF g%=137 THEN x=x+inc%
1700 IF g%=138 THEN y=y-inc%
1710 IF g%=139 THEN y=y+inc%
1720 PROCcursor
1730 ENDPROC
1740 :
1750 DEF PROCcursor
1760 MOVE x-15,y:DRAW x+15,y
1770 MOVE x,y-15:DRAW x,y+15
1780 ENDPROC
1790 :
1800 DEF PROCplanets
1810 CLS:CLG
1820 FOR n%=1 TO p%
1830 PROCdisc(Xp(n%),Yp(n%),Rp(n%))
1840 NEXT n%
```

Continued on page 27

BEEBUG WORKSHOP

Faster Graphics

by David Fell

The Beeb's graphics are better than those on most micros and you probably use them at least some of the time When you do, you quickly come up against the fact that, regrettably, there isn't a command on the model B to draw circles automatically. The obvious answer is to write a procedure to do the job and, this month, I'll show you an effective way of doing this. The technique used is applicable elswhere, so may well be of interest to Master owners too

Like lots of things in life, the "obvious" way of drawing a circle is not necessarily the best. I have listed here a very simple circle-drawing procedure.

It draws a circle with centre at (x%,y%) and radius r% graphics units. It works perfectly well, but it is terribly s-l-o-w. The main reason is that it has to work out no fewer than 72 sine and cosine functions in drawing the circle. Although BBC Basic is generally very fast, its trig calculations are not its best feature.

So, any routines which use a lot of trigonometry can be irritatingly slow. There are ways around that and here's one approach to drawing circles without calculating a lot of sines and cosines. It relies on two equations you may remember from school

```
Sin(A+B)=Sin(A)*Cos(B)+Cos(A)*Sin(B)
Cos(A+B)=Sin(A)*Sin(B)-Cos(A)*Cos(B)
```

```
SIMPLE CIRCLE
 10000 DEF PROCcircle(x%,y%,r%)
 10010 LOCAL ang,steps%,xtemp%,ytemp%
 10020 steps%=36
 10030 MOVE x%,y%+r%
 10040 FOR ang=2*PI/steps% TO 2.01*PI
       STEP 2*PI/steps%
 10050   xtemp%=x%+r%*SIN(ang)
 10060   ytemp%=y%+r%*COS(ang)
 10070   DRAW xtemp%,ytemp%
 10080   NEXT
 10090 ENDPROC
```

Given the values of sine and cosine for a starting angle and for the fixed angle we want to step in, these equations make it easy to calculate a whole series of trig functions. The procedure above uses steps of 10 degrees and calculates sines and cosines for 10, 20, 30 degrees, etc. But:

```
Sin(10)=Sin(0+10)
Sin(20)=Sin(10+10)
Sin(30)=Sin(20+10)
Sin(40)=Sin(30+10)
etc.
```

Look at the equations above. If we know the sine and cosine of 0 and 10 degrees, we can calculate the values for 20 degrees. Having got there, 30 degrees follows, and then 40 degrees So it goes - calculate the starting point and the

increment (10 degrees in this case) and, from then, we need use only simple arithmetic to find the sine and cosine for every step up to 360 degrees, or as far as we want. It's not a very good way of finding, say, just Sin(257), but it's perfect for a regular series of calculations, and much faster than using the built-in trig functions.

If you're still with me, here's how to use this trick to draw a circle. It does the same job as PROCcircle, plus a bit more. If *fill%* is TRUE, then it will draw a filled circle (i.e. a solid disc), rather than a line. It also speeds things up by only calculating a quarter of the circle and drawing the 4 quadrants simultaneously.

Lines 10020-10040 choose the number of steps in drawing the circle - big ones need more accuracy than little ones. Lines 10050 and 10060 calculate the fixed trig increments to use in the equations and line 10080 sets the initial values of Sin(0) and Cos(0). The *xo1%*, *yo2%*, etc., are used to record the progress of the 4 arcs.

```
FAST CIRCLE
10000 DEF PROCfastcirc(x%,y%,r%,fill%)
10010 LOCAL a%,cinc,cos,plotno%,sin,
      sinc,stemp,steps%,xo1%,xo2%,
      xtemp%,yo1%,yo2%,ytemp%
10020 steps%=32
10030 IF r%>300 THEN steps%=40
10040 IF r%<50 THEN steps%=20
10050 sinc=SIN(2*PI/steps%)
10060 cinc=COS(2*PI/steps%)
10070 IF fill% THEN plotno%=85 ELSE
      plotno%=5
10080 sin=0:cos=1
10090 xo1%=x%:yo1%=y%+r%
10100 xo2%=x%:yo2%=y%-r%
10110 xo3%=x%:yo3%=y%+r%
10120 xo4%=x%:yo4%=y%-r%
10130 FOR a%=1 TO steps% DIV 4
10140   stemp=sin
10150   sin=sin*cinc+cos*sinc
10160   cos=cos*cinc-stemp*sinc
10170   xtemp%=r%*sin:ytemp%=r%*cos
10180   MOVE xo1%,yo1%
10190   IF fill% THEN MOVE x%,y%
10200   xo1%=x%+xtemp%:yo1%=y%+ytemp%
10210   PLOT plotno%,xo1%,yo1%
10220   MOVE xo2%,yo2%
10230   IF fill% THEN MOVE x%,y%
10240   xo2%=x%-xtemp%:yo2%=y%-ytemp%
10250   PLOT plotno%,xo2%,yo2%
10260   MOVE xo3%,yo3%
10270   IF fill% THEN MOVE x%,y%
10280   xo3%=x%-xtemp%:yo3%=y%+ytemp%
10290   PLOT plotno%,xo3%,yo3%
10300   MOVE xo4%,yo4%
10310   IF fill% THEN MOVE x%,y%
10320   xo4%=x%+xtemp%:yo4%=y%-ytemp%
10330   PLOT plotno%,xo4%,yo4%
10340   NEXT
10350 ENDPROC
```

Lines 10140-10160 are the important ones, actually calculating the sines and cosines by the equations above. The rest of the FOR loop is concerned with drawing the 4 arcs plus, if necessary, filling in the segments to the circle's centre to make a solid disc.

To give you an idea of the routine's speed, it draws circles about 3.5 times faster than PROCcircle. Because PLOT85 is inherently slow, it is not quite so good at discs - its speed drops to around 1.8 times. Overall, though, circles are drawn in fractions of a second - try it.

The best way to find out what the procedure can and can't do is to play with it, and to incorporate it in your own programs. However, here is a demo program to give you an idea of the routine's speed:

```
100 MODE 2
110 REPEAT
120   *FX 15,0
```

```
130   X%=200+RND(880)
140   Y%=200+RND(640)
150   R%=10+RND(400)
160   GCOL RND(4)-1,RND(8)-1
170   PROCfastcirc(X%,Y%,R%,TRUE)
180   UNTIL INKEY$(0)=" "
190 END
```

Enter that short program along with PROCfastcirc. When you run it, it draws randomly-positioned discs of random size, in random colours. The process continues until you press the space bar.

This technique of rapidly calculating successive trig values for steadily increasing angles can be used in all sorts of calculations. Although it is most useful for graphics, there are lots of other applications in which it can help.

You might like to think how to extend PROCfastcirc to draw ellipses. A circle is just a special form of ellipse. For starters, try an ellipse with major axis (the long one) length *2a* and minor axis *2b*, centred at (*centrex,centrey*). Its equations are:

```
x=centrex+(a*COS(ang))
y=centrey+(b*SIN(ang))
```

When you've done that, how about drawing ellipses at angles? It's not that difficult. Next month, I'll show you how to speed-up the calculation of maths functions.

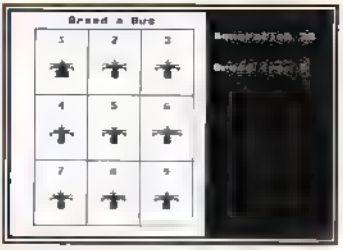
Note: this Workshop is based on one first published in BEEBUG Vol.4 No.4.

Gravity and Orbits (continued from page 24)

```
1850 ENDPROC
1860 :
1870 DEF PROCdisc(Xc,Yc,r)
1880 GCOL0,1
1890 Ca=COS(PI/40):Sa=SIN(PI/40)
1900 CA=1.SA=0.MOVE r*CA+Xc,r*SA+Yc
1910 FOR A=1 TO 80
1920 Cp=CA*Ca-SA*Sa
1930 CA=Cp*Ca-Sp*Sa:SA=Sp*Ca+Cp*Sa
1940 x=r*CA+Xc:y=r*SA+Yc
1950 MOVE Xc,Yc:PLOT 85,x,y
1960 NEXT A
1970 GCOL3,1
1980 ENDPROC
1990 :
2000 DEF PROCinfo
2010 *FX11
2020 y$=CHR$131:c$=CHR$134:w$=CHR$135
2030 PRINT TAB(12,2)y$+"GRAVITY MAP"
2040 PRINT''c$+"You can define up to 7
planets"
2050 PRINT c$+"by centre and radius."
2060 PROCselectp
2070 PROCselectc
2080 PRINT''w$+"Press any key to start.
"
2090 PRINT'w$+"After mapping is complet
e:"
2100 PRINT w$+"Press R to Repeat for an
other layout,"
2110 PRINT w$+"Press Q to Quit."
2120 *FX12
2130 ENDPROC
2140 :
2150 DEF PROCselectp
2160 PRINT'w$+"How many?  ";
2170 REPEAT:p%=GET-48:UNTIL p%>0 AND p%
<8
2180 PRINT;p%
2190 ENDPROC
2200 :
2210 DEF PROCselectc
2220 *FX21
2230 PRINT''c$+"Choose whether to map t
he"
2240 PRINT c$+"gravitational field with
:"
2250 PRINT'w$+"1  Zero slope contours"
2260 PRINT w$+"2  Maximum slope lines o
f force "
2270 REPEAT:c%=GET-48:UNTIL c%>0 AND c%
<3
2280 PRINT w$+"Choice is: ";c%
2290 ENDPROC
```

Best of BEEBUG

Applications I Disc



Business Graphics for producing graphs, charts and diagrams
Video Cataloguer catalogue and print labels for your video cassettes
Phone Book - an on-screen telephone book which can be easily edited and updated
Personalised Letter-Headings design a stylish logo for your letter heads
Appointments Diary - a computerised appointments diary
Mapping the British Isles - draw a map of the British Isles at any size
Selective Breeding a superb graphical display of selective breeding of insects
The Earth from Space draw a picture of the Earth as seen from any point in space
Personalised Address Book - on-screen address and phone book
Page Designer a page-making package for Epson compatible printers
World by Night and Day a display of the world showing night and day for any time and date of the year

File Handling for All

on the BBC Micro and Acorn Archimedes

by David Spencer and Mike Williams

Computers are often used for file handling applications yet this is a subject which computer users find difficult when it comes to developing their own programs. *File Handling for All* aims to change that by providing an extensive and comprehensive introduction to the writing of file handling programs with particular reference to Basic.

File Handling for All, written by highly experienced authors and programmers David Spencer and Mike Williams, offers 144 pages of text supported by many useful program listings. It is aimed at Basic programmers, beginners and advanced users, and anybody interested in File Handling and Databases on the Beeb and the Arc. However, all the file handling concepts discussed are relevant to most computer systems, making this a suitable introduction to file handling for all.

The book starts with an introduction to the basic principles of file handling, and in the following chapters develops an in depth look at the handling of different types of files e.g. serial files, indexed files, direct access files, and searching and sorting. A separate chapter is devoted to hierarchical and relational database design, and the book concludes with a chapter of practical advice on how best to develop file handling programs.

The topics covered by the book include

Card Index Files, Serial Files, File Headers, Disc and Record Buffering, Using Pointers, Indexing Files, Searching Techniques, Hashing Functions, Sorting Methods, Testing and Debugging, Networking Conflicts, File System Calls

The associated disc contains complete working programs based on the routines described in the book and a copy of Filer, a full-feature Database program originally published in BEEBUG magazine

ASTAAD

Enhanced ASTAAD CAD program for the Master, offering the following features:

- full mouse and joystick control
- built-in printer dump
- speed improvement
- STEAMS image manipulator
- Keystrips for ASTAAD and STEAMS
- Comprehensive user guide
- Sample picture files

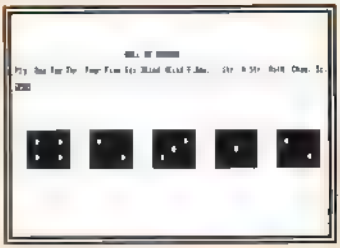
	Stock Code	Price		Stock Code	Price
ASTAAD (80 track DFS)	1407a	£5.95	**ASTAAD** (3.5" ADFS)	1408a	£5.95
Applications II (80 track DFS)	1411a	£4.00	**Applications II** (3.5" ADFS)	1412a	£4.00
Applications I Disc (40/80T DFS)	1404a	£4.00	**Applications I Disc** (3.5" ADFS)	1409a	£4.00
General Utilities Disc (40/80T DFS)	1405a	£4.00	**General Utilities Disc** (3.5" ADFS)	1413a	£4.00
Arcade Games (40/80 track DFS)	PAG1a	£5.95	**Arcade Games** (3.5" ADFS)	PAG2a	£5.95
Board Games (40/80 track DFS)	PBG1a	£5.95	**Board Games** (3.5" ADFS)	PBG2a	£5.95

All prices include VAT where appropriate. For p&p see Membership page.

RISC Developments Ltd, 117 Hatfield Road, St Albans, Herts AL1 4JS.

Best of BEEBUG

Board Games



Solitaire - an elegant implementation of this ancient and fascinating one-player game and a complete solution for those who are unable to find it for themselves.

Roll of Honour Score as many points as possible by throwing the five dice in this on-screen version of 'Yahtze'

Patience - a very addictive version of one of the oldest and most popular games of Patience.

Elevenses - another popular version of Patience lay down cards on the table in three by three grid and start turning them over until they add up to eleven.

Cribbage - an authentic implementation of this very traditional card game for two, where the object is to score points for various combinations and sequences of cards.

Twiddle a close relative of Sam Lloyd's sliding block puzzle and Rubik's cube where you have to move numbers round a grid to match a pattern

Chinese Chequers a traditional board game for two players, where the object is to move your counters, following a pattern, and occupy the opponent's field.

Aces High another addictive game of Patience where the object is to remove the cards from the table and finish with the aces at the head of each column.

Applications II Disc

Crossword Editor for designing, editing and solving crosswords

Monthly Desk Diary - a month-to-view calendar which can also be printed

3D Landscapes generates three dimensional landscapes

Real Time Clock - a real time digital alarm clock displayed on the screen

Running Four Temperatures calibrates and plots up to four temperatures

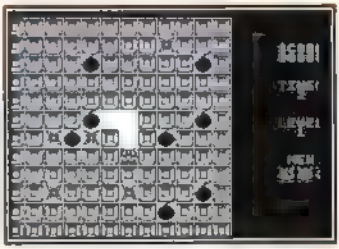
Julia Sets - fascinating extensions of the Mandelbrot set

Foreign Language Tester - foreign character definer and language tester

Share Investor - assists decision making when buying and selling shares

Label Processor for designing and printing labels on Epson compatible printers

Arcade Games



George and the Dragon Rescue Hideous Hilda from the flames of the dragon but beware the flying arrows and the moving holes on the floor

Ebony Castle - You, the leader of a secret band, have been captured and thrown in the dungeons of the infamous Ebony Castle Can you escape back to the countryside, fighting off the deadly spiders on the way and collecting the keys necessary to unlock the coloured doors?

Knight Quest - You are a Knight on a quest to find the lost crown, hidden deep in the ruins of a weird castle inhabited by dangerous monsters and protected by a greedy guardian.

Pitfall Pete - Collect all the diamonds on the screen, but try not to trap yourself when you dislodge the many boulders on your way

Builder Bob Bob is trapped on the bottom of a building that's being demolished. Can you help him build his way out?

Minefield Find your way through this grid and try to defuse the mines before they explode, but beware the monsters which increasingly hinder your progress.

Manic Mechanic - Try to collect all the spanners and reach the broken-down generator before the factory freezes up.

Quad - You will have hours of entertainment trying to get all these different shapes to fit

	Stock Code	Price		Stock Code	Price
File Handling for All Book	BK02b	£9.95			
File Handling for All Disc (40/80T DFS)	BK05a	£4.75	**File Handling for All** Disc (3.5" ADFS)	BK07a	£4.75
Joint Offer book and disc (40/80T DFS)	BK04b	**£11.95**	**Joint Offer book and disc** (3.5" ADFS)	BK06b	**£11.95**
Magscan (40 DFS)	0005a	£9.95	**Magscan Upgrade** (40 DFS)	0011a	£4.75
Magscan (80T DFS)	0006a	£9.95	**Magscan Upgrade** (80T DFS)	0010a	£4.75
Magscan (3.5" ADFS)	1457a	£9.95	**Magscan Upgrade** (3.5" ADFS)	1458a	£4.75

All prices include VAT where appropriate. For p&p see Membership page.

512 Forum

by Robin Burton

The main item this month results from a reader's letter, which also reminded me of comments David Harper made when he sent the latest copy of PCCE to me

USEFUL SOFTWARE?

A while ago John Sutton wrote to me with his observations on the programs from 28 cover discs of various PC magazines over the past couple of years. In fact there's so much information that I can't see how to use it directly here, and of course it relates to magazines which you either already have, or you don't. However, it does prompt me to mention magazines as one cheap potential source of DOS software which some of you might not have tried.

You probably remember that (8-bit) BBC micro magazines included a cover disc from time to time, although it was a pretty rare event. Moreover, however infrequent they were they were never regular, I'd guess partly because of cost, but equally because of content.

8 bit BBC magazine cover discs invariably contained either a demonstration of a new application or, more likely, a cut-down version of a game. These were obviously promotions, intended to tempt the reader into buying the full version of the software, usually at full price. Without doubt the software publishers contributed most if not the entire cost of the disc, since it was really just another form of advertising.

While a free preview or demonstration of software is no bad thing if you happen to be interested in that particular (type of) program, I'd guess that few users really found these disc of much interest. However, you did at least gain a free disc out of the deal if nothing else.

In short, cover discs on BBC micro magazines were of limited interest or use despite their rarity. They never contained fully working software as I recall, and no publisher ever made the mental leap from a promotional disc to a cover disc containing, for example, reader supplied usable and useful utilities, though BEEBUG has always supplied a magazine disc as an optional item.

In many ways this mirrors the fact that, despite the number of 8-bit BBC micros sold (estimates vary, but over 1.5 million is a conservative figure) neither shareware nor PD got off the ground until very late in the machine's life. I reckon that Alan Blundell had his work cut out starting his BBC PD operation and he deserves all the support he gets, especially now that some 512 discs and ex-commercial programs are included.

PC MAGAZINES

In contrast to BBC micro magazines, the majority of PC magazines nowadays do include a cover disc, many of them every month and in some cases two discs are provided. Of course, the use of the word 'free' is open to debate, but you can't complain if a magazine offers a cover disc (or two), has more pages than Acorn User or Micro User have ever had (combined!) and still costs no more than either of them did several years ago.

There are good reasons for the difference, so we shouldn't judge Redwood or Europa too harshly. PC

magazine publishers aren't just being altruistic, they have a much larger circulation than BAU or MU ever did You'll also find that out of, say, 400 pages half is advertising space in most of them. Volume production means low unit costs anyway, but add high advertising revenues too and it equals a low cover price without pain. Those are the economics, but all the reader need do is be pleased with the situation and take advantage of it.

I haven't surveyed the market for you, a visit to any large newsagent will accomplish that, but a good example is *Computer Shopper*. (No, I have no connection with it except as an occasional purchaser.) 'Shopper' doesn't always have a cover disc, but it does always include sections for Archimedes, Amiga, Atari and Macintosh as well as PCs, yet only this year it increased its price from £1.20 to £1.49. How dreadful!

PC Direct also sells at a similar price, but some publications are distinctly more profit oriented. At the other end of the scale are magazines costing twice that amount and more. One that comes to mind, although it does have a cover disc and occasionally hits 500 pages, has an advertising content just as high as any but only glossier paper to justify its stunning price. I suggest that neither the thrifty nor the environmentally concerned should be amongst its customers.

A quick look at the contents, and one or two judicious test purchases, will soon show which magazines are the best value, are the most interesting, and are the most likely to offer useful programs for your 512. There are occasionally demos on PC cover discs just as there were on BBC discs, but just as often a few and sometimes 50 or more small DOS utilities will appear on a disc.

One problem for most 512 users is that most PC cover discs are 3.5 inch these days Some magazines are still published with both 3.5 and 5.25 inch discs and you buy the one that suits, though note that 5.25" discs are always 360K while 3.5" discs are 720K, so the two don't always provide the same (or the same quantity of) software in any given issue. Others are produced only with 3.5" discs, but you can return the disc to swap it for a 5.25", although that's obviously less convenient.

COMPATIBILITY POINTERS

With regard to the programs you'll find on PC cover discs, overall the range of problems is typical for the 512, but some general guidelines will help a bit.

All executable PC programs are one of two type, .COM or .EXE (there are no .CMD files in MS-DOS) and this is the first indicator. Both types contain machine code, but they're quite different internally and in how they were produced. A bit more explanation will help you to see why this is and how it can affect 512 compatibility. Naturally these points can be applied to programs from other sources too.

COM programs are written in assembly language which is directly converted into a program file by a single operation. Note that direct program assembly always produces a COM file, even if the source code is in several separate files and/or uses a library. Because of this one-shot approach, COM files must be less than 64K in size, although they can access any memory in the machine and expand internal storage as much as they like once they're running.

An assembler's job is only to convert source code directly into machine code by a straightforward look-up procedure, without adding anything and without making any assumptions. As a result, if

the programmer didn't write the instructions in the source it doesn't happen in the program. There are no 'pre-packed' facilities in assembly code, so external functions are normally accessed via standard DOS interrupts This is simple, fast and memory efficient, prime reasons for using an assembler in the first place

In general most COM programs are written with XT computers in mind, and since these, like the 512, have limited memory and run only DOS, COM programs are a good bet for the 512. One or two magazines definitely tend to cater for this market as much or more than Windows, and so they're always worth checking when you pass the magazine racks

Magazine cover disc COM programs will be utilities, covering a wide range of purposes, some it must be said pretty obscure, but always remember that disc utilities are generally the least reliable sort of program for the 512

Despite potential disc problems, and the fact that some programs seem to be answers looking for questions, one or two gems appear from time to time. For example, a while ago I came across a program called CODE which I find extremely useful. You can supply either a telephone exchange name when you call CODE, in which case it will return the STD dialling code for that exchange, or you can supply a dialling code and it will tell you the exchange it belongs to. CODE is simple and neat, when I have a dialling code I want to identify, it's just about the perfect tool for the job.

By contrast, EXE programs, even if written for DOS, are much less likely to run in the 512 than COM programs. In detail there are a large number of causes of potential problems in the 512 but, graphics apart, troubles can, in general, be explained as follows.

Producing an EXE file isn't a single process. Various modules or sections of the final program can be and often are written separately, sometimes using different languages. These modules (even if there is only one) are first assembled or compiled into separate object modules. This produces one or several lumps of machine code, but these aren't yet programs, they must be converted into an executable program file which is done by a linker. This, as its name implies, links the various raw pieces of code together to form the finished program

If all the source modules are written in assembler it's probable that the resulting program will be as likely to run in the 512 as a COM file. However, since EXE programs can be larger than COM files and not many people have assembler expertise, source code is usually written in a high-level language such as C. This is easier and faster to write in than assembler, in part because I/O functions in high level languages are provided by libraries of 'standard' routines which can be called, such as STDIO (standard I/O) in C. These routines are written by the compiler's authors, not the program's, and are added into the program at compile time

Unfortunately, the subroutine libraries of many recent PC compilers, regardless of language, either rely heavily on, or make assumptions about the environment in which they run and will produce programs that display similar characteristics. For example, rather than using DOS interrupts for keyboard input or printer output, both of which are slow, but they all work even in the 512, some languages' library routines insist on reading or writing ports directly

The truth is that these devices are slow too, so in fact the DOS functions are

perfectly adequate for most purposes and peeking or poking memory-mapped I/O gains nothing. Even so, it's a common technique and if that's how a language's library routines work that's how all the programs compiled by that language will work too, with the result that they won't work in a 512 Knowing this doesn't make life any easier, but it's important to remember it's not usually the application itself that's the cause of troubles, it's the way it was produced. This is why compatibility is a more common problem with recent versions of some EXE programs.

Finally, leaving program types and getting back to magazine discs, it hardly needs saying, but I'd better - any magazine (disc) aimed primarily at Windows will be a waste of time and money for 512 users.

Despite some problems, many PC magazines do offer another worthwhile source of programs for the 512. You can not only keep up to date with latest PC developments, you can have a good read and perhaps gain a useful utility or two by spending only a couple of pounds or so.

MORE 512 SPECIALS

I don't yet know how popular the software offer in last month's Forum will be, but I've decided to do it again and throw in three of my own programs which, like David's were originally written for my own convenience.

This time you get a disc copying program called FASTBACK which, unlike PC programs does handle 800K discs (plus the other sizes) and which will duplicate a disc as fast as the 512 can do it by using all the available memory for both reads and writes. By comparison DISK is 50% slower

Accompanying FASTBACK are two other programs. The second is SAK (Seek And Kill) which locates any file or files matching a given name (which can be wildcarded) in any directory on any drive. For each occurrence of a matching name you have the option of deletion. This can be very useful if you sometimes find you have a number of redundant *.TMP, *.BAK or similar files hanging around wasting disc space.

The third program is DESTROY, which likewise accepts a (wildcarded) name and offers deletion, but this time the matched object can be either a file or a directory, in the latter case including any subdirectories and files. One DESTROY command can replace any number of separate DELs, CDs and RDs and effectively also offers a way of wiping a complete disc clean which is very much faster than reformatting

Both SAK and DESTROY can remove system and hidden files directly, but both are totally safe because positive confirmation is needed before either deletes anything. Also, to delete directories and all their contents in DESTROY you really have to insist that it does so.

You won't need these programs every day, but when you do they can save a lot of effort and time. The cost's the same as before, £2.00, so at 60p a program it doesn't take much to justify them. As before, write to me c/o Essential Software if you'd like them.

If anyone else has a masterpiece of code that they think might be useful to fellow 512 users (and which is all your own work) send me a copy and, if I agree with you and can accumulate two or three of them there'll be another offer

Next month I'll finally get round to PKZIP.

B

Sound (3)

Alan Wrigley continues his description of the BBC micro's sound system.

ENVELOPES

The previous two articles on the sound system have been concerned with the use of the Basic keyword SOUND The sounds we have produced so far have varied in pitch, amplitude and duration, and we have also introduced an element of noise. The one main characteristic of a sound so far missing is its envelope, and that is what we are going to look at this month.

An envelope describes how the amplitude of a sound (and/or its pitch as we will see later) changes over the duration of the sound. An envelope can therefore be thought of as a graph of amplitude against time, and is often represented in this form in diagrams. Figure 1 shows a typical envelope. Here the amplitude of the sound rises sharply to its peak, then falls off rapidly until it reaches a kind of plateau, from where it decays more slowly until at some point it is terminated fairly abruptly.

Figure 1. A Typical amplitude envelope

A note played on a piano is of this type. When you hit a key, there is a sharp burst of sound, and then the note fades away slowly until the key is released, whereupon it will die fairly quickly. Of course, in real life the envelope will be more complex than this - the amplitude will be unlikely to rise and fall in neat straight lines. Nevertheless, a good approximation to a natural envelope can be made by this method, and so it lends itself very well to computerisation

This type of envelope is known as an *Attack-Decay-Sustain-Release* (or ADSR) envelope, reflecting the four phases of the envelope. The attack phase describes the increase in amplitude from zero to its maximum level; the decay phase is the period during which the amplitude falls rapidly to a more sustainable level; the sustain phase refers to the period of slower decay (or in many cases completely stable amplitude); and the release phase describes the final termination period of the sound. Synthesisers normally allow the characteristics of all these phases to be set independently, thus enabling many different kinds of sound to be imitated.

BBC Basic provides a powerful ENVELOPE command which allows you to do the same for sounds produced using the SOUND command The use of ENVELOPE is quite complex, and you may well find that you need to experiment a great deal in order to achieve the results you want.

THE ENVELOPE COMMAND IN DETAIL

Normally up to four different envelopes can be defined at any one time. These are numbered 1 to 4, and the envelope

number is given as part of the ENVELOPE command's parameters, as we will see in a moment. Before an envelope will have any effect, however, we must tell the computer when a SOUND command is issued that we want it to be controlled by an envelope. This is very easy - since the envelope describes the amplitude characteristics of the sound, we do not need the A (amplitude) parameter, so we replace it with the envelope number. If you remember from the previous articles, this parameter was always zero or negative when referring directly to amplitude; by making the parameter positive we can specify the envelope number instead (1-4). For example, consider the following lines:

```
SOUND 1,-15,100,5
SOUND 1,2,100,5
```

In the first line, the sound is produced at maximum volume; in the second, the amplitude of the sound is regulated by envelope 2.

The ENVELOPE command itself is followed by 14 parameters, so take a deep breath and read on. Because of the command's complexity, the rest of this month's article will be devoted to an explanation of the parameters. This means a lot of theory, I'm afraid, but next month we will go through the process of developing some actual examples. If you don't understand all the theory at first, re-read the parts that are puzzling you and experiment with the command as much as you can.

The parameters are as follows

```
ENVELOPE N,T,PI1,PI2,PI3,PN1,PN2,PN3,
          AA,AD,AS,AR,ALA,ALD
```

Figure 2 gives a brief description of each parameter, and they will now be explained in more detail

Parameter	Range	Function
N	1 to 4	Envelope number
T (bits 0-6)	1 to 127	Length of each step in centiseconds
T (bit 7)	0 or 1	0 = auto-repeat pitch envelope 1 = don't repeat pitch envelope
PI1	128 to 127	Change of pitch per step in section 1
PI2	-128 to 127	Change of pitch per step in section 2
PI3	-128 to 127	Change of pitch per step in section 3
PN1	0 to 255	Number of steps in section 1
PN2	0 to 255	Number of steps in section 2
PN3	0 to 255	Number of steps in section 3
AA	-127 to 127	Change of amplitude per step during attack phase
AD	-127 to 0	Change of amplitude per step during decay phase
AS	-127 to 127	Change of amplitude per step during sustain phase
AR	-127 to 0	Change of amplitude per step during release phase
ALA	0 to 126	Target amplitude level at end of attack phase
ALD	0 to 126	Target amplitude level at end of decay phase

Figure 2 Parameters to the ENVELOPE command

N specifies the envelope number. This should match the number given in the SOUND command which is to be controlled by the envelope. Four envelopes (1-4) are available as standard, but if your program doesn't make use of the Basic file-handling statement BPUT# at all, then up to 16 envelopes (1-16) may be defined.

To help you tailor the envelope to your own needs as accurately as possible, each phase can be broken down into a number of steps of equal duration. As you can see from Figure 2, many of the other parameters make use of this. For example, with AA you can determine how sharp the attack gradient is by specifying the amplitude increase per step. The bottom 7 bits of the T parameter (bits 0-6) enable you to set

the step length (in centiseconds) which will be used over the entire envelope. Since this value will be used by *all* the other parameters that relate to the step length, you should choose it with care You may need to experiment with different step lengths until the sound is right

Clearly the lengths of the attack and decay phases are determined by the step length (T), the volume change per step (AA and AD) and the number of steps required to reach the target volume (ALA and ALD). For example, if T is 1, AA is 1 and ALA is 100, the attack phase will last 1 second. Similarly, the length of the release phase is determined by the number of steps required to get to zero from the level at the end of the sustain phase However, the length of the sustain phase itself cannot be determined solely by the envelope parameters. This is where the D (duration) parameter of the SOUND command comes in. This specifies the length of the sound from the start of the attack to the end of the sustain. Thus the sustain length will be duration-(attack+delay). Note that if D is shorter than the attack or attack-plus-delay phases, the sound will be terminated before it reaches the sustain level

I mentioned earlier that an envelope can describe pitch as well as amplitude. Both types of envelope are covered in one ENVELOPE command, the pitch element being determined by PI1 - PN3. If you don't want the pitch of the sound to change while it is playing, then PI1 - PN3 should all be set to zero. However, many sounds *do* alter their pitch over time (a siren, for example), and these parameters allow you to cater for this. The pitch envelope consists of three sections; for each of these you can set the number of steps (using PN1 - PN3) and the change of pitch per step (PI1 - PI3), and the values for each section are independent of each other.

The pitch envelope is also independent of the amplitude envelope. This is important, since for some sounds the two may coincide in time, while for others they may not. An example of the first might be a cat miaowing - the pitch tends to rise and fall with the amplitude of the miaow. An example of unsynchronised envelopes would be a police siren approaching and then disappearing into the distance - here the pitch of the siren changes repeatedly between the two levels while the amplitude slowly increases and then decreases (unfortunately the Doppler effect can't be achieved within a single sound - there have to be *some* limitations!).

In the case of the siren, the pitch envelope repeats over and over again for the duration of the sound. In other cases you may want the pitch envelope to take effect once only at the start of the sound, after which the pitch remains constant. This is determined by the top bit of the T parameter. By default this is zero, meaning the pitch envelope auto-repeats, but if you set it to 1 then it doesn't repeat.

Next month I will look at the remaining parameters and describe some examples of the use of ENVELOPE. In the meantime, you might like to try out the following example:

```
10 ENVELOPE [illegible]
[illegible]
[illegible]
```

All will be explained next month!

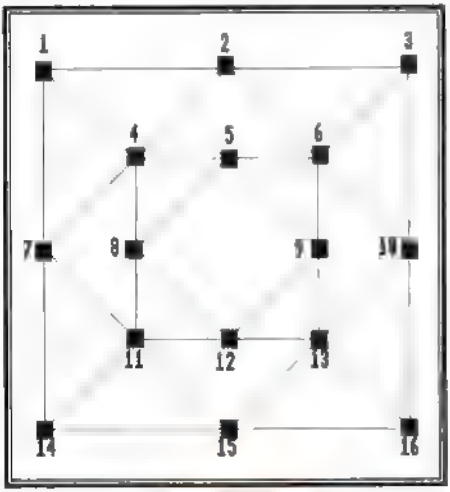
More Hide and Seek

Marshal Anderson looks at more complex game structures.

If you went Snib hunting last month you should by now have sent a fair few to meet their programmer. This month I want to look at something a little more complex but, again, the idea is to give you a basic game which you can jazz up in all sorts of weird and wonderful ways.

THIS TIME IT'S GRIBBITS

The Gribbit is not quite as aggressive as the Snib in as much as it isn't hunting you (usually). It sleeps in a network of caves a plan of which you see below.

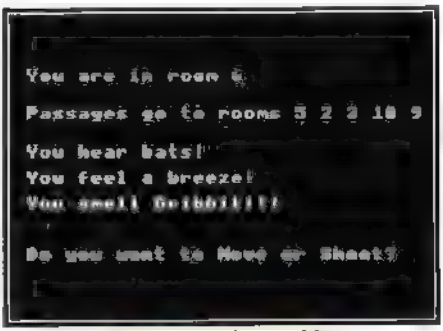


A cunning plan

Type in the listing below and, using the plan, have a go at hunting down the Gribbit - then try it without the map. Some care and strategy will bring you success every time, with levels 1 and 2, but level 3 should be quite a challenge. The specific differences between the levels are discussed below.

In programming terms we have to deal with this quite differently from the Snub grid; each cave has to have its links to other caves specifically stated. This is done in the DATA statements in lines 2470-2620. The order of the statements reflects their cave number so line 2470 is the data for cave 1. To make life easy, the data of each cave is padded out with zeros to 5 items, the maximum number of links for any cave (see the map above). Every time you wanted to know the links for a cave you could do a READ but this is rather slow so the whole lot is read into an array in *PROCinitialise* which also sets up the *HAZARDS%* array.

Once we've initialised things and printed out the instructions with *PROChello*, we need to set up the actual game. *PROCstartnewgame* first gets the difficulty level in *DIFF%*. Next it sets up the pit traps (to fall into), giant bats (to carry you around), the Gribbit's position and your starting point in case you want to play the same game again. If *DIFF%* is 1 line 2360 makes two of the *HAZARD%()* elements -1, one bat and one pit, so they don't appear in any of the rooms. This is the game at its easiest.



Now you're in trouble

In the actual game the first procedure we use is *PROCwhereami* which prints your position and the rooms you can move to,

the variable *YOUAREHERE%* keeps track of your position. Once we know where we are we need to check if any of the hazards are in adjoining rooms - *PROCanynews* does this. What it is looking for is a match between any of the cave numbers in *HAZARD%()* and the caves you could move into. In a purely arbitrary way elements 1 and 2 are used for the bats, 3 and 4 for the pits and 5 for the Gribbit. For each match it finds it prints a message. If there are two sets of bats or pits the respective message is printed twice.

Obviously it's a Big Clue if you see two lots of bats or pits from one room; this is where difficulty level 3 comes in. The bit of code at the end of lines 1480 and 1490 will make sure that, even if there are two lots of bats or pits there will only be one message printed making their locations more difficult to guess.

Now we know where we are and what is in the caves around us we get the choice of moving or shooting. *FNwhatnow* gets the choice here using GET$ and accepts upper and lower case. Depending on the reply *PROCmove* or *PROCshoot* are called and they both use INPUT to get the relevant cave number Cave numbers are checked to make sure they are valid and lines 1170 and 1570 check for a Return with no number typed - this avoids a move into room 0.

If we're moving and the move is legal then *YOUAREHERE%* is updated; if we're shooting then things get more complicated PROCshoot also checks for a valid cave to shoot into - only adjacent ones are acceptable - and it then checks HAZARDS%(5) to see if you have hit the Gribbit. If you have, *PROCgotim* displays a message and *PLAYING%* is set to FALSE so the game will drop out of the loop. If you miss the target things get a bit more interesting: from a game design angle there's no point in just letting your player shoot into all the surrounding caves when they smell the Gribbit - that's no game at all. What we do is move the Gribbit if they miss, possibly into the cave they're in, so there's a real incentive to work things out strategically.

Having moved or shot, there is a check on the new location in *PROChowami*. This shifts you about if you've got bats, and double checks the new location so you can get more bats, drop down a hole or meet the Gribbit. If you go down a hole or meet the Gribbit it ends the game.

SO, WHERE NEXT?

The basic game already has three levels of difficulty; the more hazards there are to be avoided, and the less information you give about them, the more difficult the game is. Moving the Gribbit more often will certainly spice things up, you could move it each time you come across the bats or you could shift it after a certain number of moves to make the player get their act together If you were feeling really nasty you could change its position at totally random times.

The next thing is cave system design The more connections there are to each cave the more difficult it is to play the game. Linkages between caves needn't be two way or in any way logical and they could even change as the game goes on (rockfalls?), you could also write a procedure to construct totally random systems. You could try multiple Gribbits, have things that chase you and catch up if you stay in the same area too long, have torch batteries that run out so that new ones have to be found, there's plenty of scope. In fact, there's no reason why you should not develop this into full blown adventure game. There's also all the ideas about presentation that I wrote about last month.

While you're developing and testing all this you will probably find *PROCinfo* useful. Presently it just prints out the contents of *HAZARD%()* and you bung it

somewhere useful in the game loop, it always helps to know what's really going on - at least that's what my dear mum used to tell me - so give it a go and let us see the results.

```
   10 REM Program Gribbit
   20 REM Version B1 0
   30 REM Author  Marshal Anderson
   40 REM BEEBUG  July 1993
   50 REM Program subject to copyright
   60
  100 MODE 1
  110 PROChello
  120 PROCinitialise
  130 PROCstartnewgame:CLS
  140 PROCwhereamI
  150 PROCanynews
  160 IF FNwhatnow="M" PROCmove ELSE PRO
Cshoot
  170 PROChowamI
  180 IF PLAYING% GOTO 140
  190 ON FNished GOTO 130,200,210
  200 PLAYING%=TRUE:YOUAREHERE%=HAZARD%(
6):CLS:GOTO 140
  210 PRINT''''"Byeeeeeeeee"
  220 END
  230 :
 1000 DEF FNished
 1010 PRINT''''"You can:"''''"1) Hunt ano
ther Gribbit"''''"2) Hunt the same Gribbi
t"''''"3) Go home."
 1020 INPUT ''''"Enter your choice";CHOIC
E%
 1030 =CHOICE%
 1040 :
 1050 DEF PROChello
 1060 CLS
 1070 PRINT"You are about to hunt these
caves for a rabid Gribbit"''"You are arm
ed with an infinite number ofnot-very-in
telligent personal missiles"
 1080 PRINT''"You hunt the Gribbit by its
 unpleasant  smell, you will smell it wh
en it's one  room away."
 1090 PRINT''"You will be asked if you wi
sh to Move orShoot. You will then be ask
ed which roomyou want to move or shoot i
nto."
 1100 PRINT''"If you hit the Gribbit you
win, there's plenty more where that came
 from."''"If you shoot and miss you will
wake the Gribbit and it will move - anyw
here."
 1110 PRINT''"If you are ever in the same
 room as the Gribbit you will expire fro
m the smell."''"Watch out also for giant
bats and"''"bottomless pits!"
 1120 PROCpak
 1130 ENDPROC
 1140 :
 1150 DEF PROCshoot
 1160 INPUT''"Shoot into which room";AIM
%
 1170 IF AIM%=0 GOTO 1160
 1180 CANSHOOT%=FALSE
 1190 FOR X%=1 TO 5
 1200 IF AIM%=ROOMS%(YOUAREHERE%,X%) CAN
SHOOT%=TRUE
 1210 NEXT X%
 1220 IF NOT CANSHOOT% PRINT' "The missi
les aren't that bright,"''"try again"'':P
ROCwhereamI:GOTO 1160
 1230 IF AIM%=HAZARD%(5) PROCgotim ELSE
PROCmoveim
 1240 ENDPROC
 1250 :
 1260 DEF PROCmoveim
 1270 NEWGRIB%=RND(15)+1
 1280 THESAME%=FALSE
 1290 FOR X%=1 TO 4
 1300 IF NEWGRIB%=HAZARD%(X%) THESAME%=T
RUE
 1310 NEXT X%
 1320 IF THESAME% GOTO 1270
 1330 HAZARD%(5)=NEWGRIB%
 1340 CLS:PRINT''"Missed:"''"You hear th
e Gribbet moving..  "
 1350 ENDPROC
 1360 :
 1370 DEF PROCgotim
 1380 CLS
 1390 PRINT"This Gribbit eats cordite!"'
''"Well done, oh mighty hunter."
 1400 PLAYING%=FALSE
 1410 ENDPROC
 1420 :
 1430 DEF PROCanynews
```

```
 1440 PRINT'
 1450 FOR X%=1 TO 5
 1460 FOR Y%=1 TO 5
 1470 IF HAZARD%(X%)<>ROOMS%(YOUAREHERE%
,Y%) GOTO 1510
 1480 IF X%=1 OR X%=2 PRINT''"You hear ba
ts!":IF DIFF%=3 X%=2
 1490 IF X%=3 OR X%=4 PRINT''"You feel a
breeze:":IF DIFF%=3 X%=4
 1500 IF X%=5 PRINT''"You smell Grıbbit!!
"
 1510 NEXT Y%
 1520 NEXT X%
 1530 ENDPROC
 1540 :
 1550 DEF PROCmove
 1560 INPUT'''"Move to which room";NEWROO
M%
 1570 IF NEWROOM%=0 GOTO 1560
 1580 CANMOVE%=FALSE
 1590 FOR X%=1 TO 5
 1600 IF NEWROOM%=ROOMS%(YOUAREHERE%,X%)
 CANMOVE%=TRUE
 1610 NEXT X%
 1620 IF NOT CANMOVE% PRINT'''"I think yo
u're lost, try again"'':PROCwhereamI:GOT
O 1560
 1630 YOUAREHERE%=NEWROOM%
 1640 CLS
 1650 ENDPROC
 1660 :
 1670 DEF PROChowamI
 1680 OOPS%=0
 1690 FOR X%=1 TO 5
 1700 IF YOUAREHERE%=HAZARD%(X%) OOPS%=X
%
 1710 NEXT X%
 1720 IF OOPS%=0 ENDPROC
 1730 IF OOPS%=1 OR OOPS%=2 PROCbats:GOT
O 1680
 1740 IF OOPS%=3 OR OOPS%=4 PROChole
 1750 IF OOPS%=5 PROCgotchya
 1760 ENDPROC
 1770 :
 1780 DEF PROCbats
 1790 YOUAREHERE%=RND(15)+1
 1800 PRINT''"Giant bats carry you to ro
om ";YOUAREHERE%''
 1810 ENDPROC
```

```
 1820 :
 1830 DEF PROChole
 1840 PRINT"You just fell down a very la
rge hole!"
 1850 PLAYING%=FALSE
 1860 ENDPROC
 1870 :
 1880 DEF PROCgotchya
 1890 PRINT''''"You just bumped into a Gr
ibbet!"''"Or it just bumped into you."''"I
t's bad news either way I'm afraid."
 1900 PLAYING%=FALSE
 1910 ENDPROC
 1920 :
 1930 DEF FNwhatnow
 1940 PRINT''''"Do you want to Move or Sh
oot?"
 1950 A$=GET$
 1960 IF A$="m" A$="M"
 1970 IF A$="s" A$="S"
 1980 IF A$<>"M" AND A$<>"S" GOTO 1950
 1990 =A$
 2000 :
 2010 DEF PROCinitialise
 2020 DIM HAZARD%(6)
 2030 DIM ROOMS%(16,5)
 2040 RESTORE
 2050 FOR X%=1 TO 16
 2060 FOR Y%=1 TO 5                ,'
 2070 READ ROOMS%(X%,Y%)
 2080 NEXT Y%
 2090 NEXT X%
 2100 ENDPROC
 2110 :
 2120 DEF PROCpak
 2130 LOCAL X%
 2140 PRINT'''"Press a key"
 2150 X%=GET
 2160 ENDPROC
 2170 :
 2180 DEF PROCwhereamI
 2190 PRINT"You are in room ";YOUAREHERE
 2200 PRINT'''"Passages go to rooms ";
 2210 FOR X%=1 TO 5
 2220 IF ROOMS%(YOUAREHERE%,X%)>0 PRINT;
ROOMS%(YOUAREHERE%,X%);" ";
 2230 NEXT X%
```

Continued on page 46

Very Big Numbers (Part 2)

Harry Fensom goes a step further into Big Maths.

This month's programs provide multiplication and division of integers up to several thousand digits

NOTES ON PROGRAMMING METHODS

Storage and indexing are the same as in the + and - programs you saw last month. The algorithm used for multiplication uses the usual successive additions followed by shifting of the arrays. The division is similar but uses trial subtractions, reverting to the previous partial remainder when the subtraction produces a negative result.

USING THE PROGRAMS

Enter the listings given below, save them and then run them. In both programs you will first be asked for the maximum number of digits you wish to use; keep in mind that a large value will take a longer time to process. Note that the result of a multiplication may require up to the sum of the number of digits in the two numbers entered. You will then be asked to enter the first and second numbers of the operation you want to carry out; numbers are entered as you usually do with the most significant digit first, i.e. from left to right. The number is terminated with the Return key.

The terms *multiplicand* and *multiplier* are the first and second numbers entered in a multiplication, and the *product* is the result. Similarly the *dividend* and *divisor* are the two numbers entered for division, and the *quotient* is the result. If the quotient is not an integer, i.e. not a whole number, then the *remainder* is also given. Thus the quotient times the divisor plus the remainder equals the dividend. This quotient and remainder are equivalent to DIV and MOD in ordinary Basic. Once both numbers have been entered the calculation will take place and will be displayed when complete. You may have to wait awhile for very large numbers; for example, a 5000 digit multiplication may take possibly 10 minutes, while division of the same length may take more than half an hour - not including the time it takes you to enter the numbers! However 1000 digit division may only take 2 minutes.

Next month we'll tackle the longer and more complex task of finding square roots.

Listing 1

```
   10 REM Program MOLlarg
   20 REM Version B.4.2c
   30 REM Author  Harry W Fensom
   40 REM BEEBUG  July 1993
   50 REM Program subject to copyright
   60 :
  100 INPUT"Maximum no. of digits "m%:ma
x%=m%+2:bytes%=max%/2
  110 PROCinit
  120 PROCassemble
  130 CLS
  140 ?ar%=?opd1p%:ar%?1=opd1p%?1:CALLcl
ear
  150 ?ar%=?opd2p%:ar%?1=opd2p%?1:CALLcl
ear
  160 ?ar%-?resp%:ar%?1-resp%?1:CALLclea
r
  170 ?ar%=?hd1p%:ar%?1-hd1p%?1:CALLclea
r
  180 ?ar%-?hd2p%:ar%?1=hd2p%?1:CALLclea
r
  190 ?tp1%-opd1%MOD256:tp1%?1-opd1%DIV2
56
  200 PROCenter("Multiplicand")
```

```
 210 ?tp1%=opd2%MOD256:tp1%?1=opd2%DIV2
56
 220 PROCenter("Multiplier")
 230 CALLmult
 240 PROCdisplay("Product ",hd2p%)
 250 END
 260 :
1000 DEF PROCinit
1010 DIMbu% max%,opd1% bytes%,opd2% byt
es%,res% bytes%,hd1% bytes%,hd2% bytes%
1020 DIMmc% &240
1030 opd1p%=&70:opd2p%=&72
1040 bup%=&74:tp1%=&76
1050 tp2%=&78:tp3%=&7A
1060 len%=&7C:mz%=&7E
1070 x%=&80:flag%=&82
1080 resp%=&84:cnt%=&86
1090 ar%=&88:rev%=&8A
1100 aux%=&8C:y%=&8E
1110 cy%=&90:hd1p%=&92
1120 hd2p%=&94
1130 oswrch=&FFEE
1140 ?opd1p%=opd1%MOD256:opd1p%?1=opd1%
DIV256
1150 ?opd2p%=opd2%MOD256:opd2p%?1=opd2%
DIV256
1160 ?bup%=bu%MOD256:bup%?1=bu%DIV256
1170 ?resp%=res%MOD256:resp%?1=res%DIV2
56
1180 ?mz%=bytes%MOD256:mz%?1=bytes%DIV2
56
1190 ?hd1p%=hd1%MOD256:hd1p%?1=hd1%DIV2
56
1200 ?hd2p%=hd2%MOD256:hd2p%?1=hd2%DIV2
56
1210 ENDPROC
1220
1230 DEF PROCenter(N$)
1240 PRINT''"Enter "N$" as DEC digits;"'
" MS Digit first & end with Return"
1250 ?ar%=?bup%:ar%?1=bup%?1:CALLclear
1260 I%=0
1270 REPEAT
1280 A1%=GET-48:IFA1%>79I%=I%-1:VDU127:
GOTO1280
1290 IFA1%<10ANDA1%>-1PRINT;A1%;:I%?bu%
=A1%
1300 I%=I%+1
1310 UNTILI%=max%ORA1%=-35
1320 IFI%=1GOTO130
1330 ?len%=(I%+2)MOD256:len%?1=(I%+2)DI
V256
1340 ?bup%=bu%MOD256:bup%?1=bu%DIV256
1350 CALLinpt
1360 ENDPROC
1370 :
1380 DEF PROCdisplay(N$,T%)
1390 IFN$<>""PRINT'N$" = "'
1400 ?ar%=?T%:ar%?1=T%?1
1410 CALLdisp
1420 PRINT'
1430 ENDPROC
1440 :
1450 DEF PROCassemble
1460 FORpass%=0TO2STEP2
1470 P%=mc%:[OPTpass%
1480 .inpt LDY#0
1490 LDAlen%:STAcnt%
1500 LDAlen%+1:STAcnt%+1
1510 CLC:LDAbup%.ADCcnt%:STAbup%
1520 LDAbup%+1.ADCcnt%+1:STAbup%+1
1530 .iloop LDA(bup%),Y:STA(tp1%),Y
1540 SEC:LDAcnt%:SBC#1.STAcnt%
1550 LDAcnt%+1:SBC#0:STAcnt%+1
1560 BCCiexit
1570 SEC:LDAbup%:SBC#1.STAbup%
1580 LDAbup%+1:SBC#0:STAbup%+1
1590 LDA(bup%),Y
1600 ASLA:ASLA.ASLA:ASLA
1610 ORA(tp1%),Y:STA(tp1%),Y
1620 CLC:LDAtp1%:ADC#1:STAtp1%
1630 LDAtp1%+1:ADC#0:STAtp1%+1
1640 SEC:LDAbup% SBC#1:STAbup%
1650 LDAbup%+1:SBC#0.STAbup%+1
1660 SEC:LDAcnt%:SBC#1:STAcnt%
1670 LDAcnt%+1:SBC#0:STAcnt%+1
1680 BCSiloop
1690 .iexit RTS
1700 :
1710 .disp LDY#0:LDX#0
1720 LDAmz%:STAcnt%:LDAmz%+1:STAcnt%+1
1730 .brd CLC:LDAar%:ADCcnt%:STAtp1%:ST
Atp2%
1740 LDAar%+1:ADCcnt%+1:STAtp1%+1:STAtp
```

```
2%+1
 1750 LDA(tp1%),Y:LSRA:LSRA:LSRA:LSRA
 1760 CPX#1:BEQpnt
 1770 CMP#0:BEQnxt1:LDX#1
 1780 .pnt CLC:ADC#48:JSRoswrch
 1790 .nxt1 LDA(tp2%),Y:AND#&F
 1800 CPX#1:BEQpnt2
 1810 CMP#0:BEQrpt:LDX#1
 1820 .pnt2 CLC:ADC#48:JSRoswrch
 1830 .rpt SEC:LDAcnt%:SBC#1.STAcnt%
 1840 LDAcnt%+1:SBC#0:STAcnt%+1
 1850 BITcnt%+1:BPLbrd
 1860 TXA BNEdexit
 1870 LDA#48 JSRoswrch
 1880 .dexit RTS
 1890 .
 1900 .clear LDY#0
 1910 LDAmz%:STAcnt%:LDAmz%+1:STAcnt%+1
 1920 .brc CLC:LDAar% ADCcnt%:STAtp2%
 1930 LDAar%+1:ADCcnt%+1:STAtp2%+1
 1940 LDA#0.STA(tp2%),Y
 1950 SEC:LDAcnt%:SBC#1:STAcnt%
 1960 LDAcnt%+1".SBC#0:STAcnt%+1
 1970 BITcnt%+1:BPLbrc
 1980 RTS
 1990 :
 2000 .mult LDY#0
 2010 STYtp3%:STYtp3%+1
 2020  mbc STYcnt%
 2030 .mdbc STYrev%
 2040 CLC:LDAopd1p%:ADCtp3%·STAtp1%
 2050 LDAopd1p%+1:ADCtp3%+1:STAtp1%+1
 2060 LDA(tp1%),Y
 2070 LDXcnt%:BPLmdbc2
 2080 LSRA:LSRA.LSRA·LSRA
 2090 .mdbc2 AND#&F:BEQsdg:STAaux%
 2100 .adlp STYy%:STYy%+1
 2110 LDAmz% STAx%·LDAmz%+1:STAx%+1·STYc
y%
 2120 .inm CLC:LDAopd2p%:ADCy%:STAtp1%
 2130 LDAopd2p%+1:ADCy%+1:STAtp1%+1
 2140 CLC:LDAhd1p%:ADCy%:STAtp2%
 2150 LDAhd1p%+1.ADCy%+1 STAtp2%+1
 2160 SED
 2170 CLC:LDAcy%:BEQcon1
 2180 SEC
 2190 .con1 LDA(tp1%),Y:ADC(tp2%),Y:STA(
tp2%),Y
 2200 CLD
 2210 STYcy%:BCCcon2
 2220 LDA#1·STAcy%
 2230 .con2 CLC:LDAy%:ADC#1:STAy%
 2240 LDAy%+1:ADC#0:STAy%+1
 2250 SEC:LDAx%·SBC#1·STAx%
 2260 LDAx%+1.SBC#0:STAx%+1
 2270 ORAx%:BNEinm
 2280 LDAcy%·BEQdcnd·INCrev%
 2290 .dcnd DECaux%:BNEadlp
 2300 .sdg LDA(hd1p%),Y:AND#&F
 2310 LDXcnt%:BPLsdl
 2320 ASLA:ASLA:ASLA:ASLA
 2330 .sdl PHA
 2340 CLC:LDAhd2p%:ADCtp3%:STAtp1%
 2350 LDAhd2p%+1:ADCtp3%+1:STAtp1%+1
 2360 PLA:ORA(tp1%),Y:STA(tp1%),Y
 2370 LDAmz%:STAy%:LDAmz%+1.STAy%+1
 2380 .shb SEC.LDAy%:SBC#1:STAy%
 2390 LDAy%+1:SBC#0:STAy%+1
 2400 CLC:LDAhd1p%:ADCy%:STAtp1%
 2410 LDAhd1p%+1:ADCy%+1:STAtp1%+1
 2420 LDA(tp1%),Y:PHA:AND#&F0
 2430 LSRrev%:ORArev%
 2440 RORA:RORA:RORA:RORA
 2450 STA(tp1%),Y:PLA:AND#&F:STArev%
 2460 LDAy%:ORAy%+1:BNEshb
 2470 DECcnt%:LDAcnt%:CMP#&FF:BNEnojmp
 2480 JMPmdbc
 2490 .nojmp CLC:LDAtp3%:ADC#1:STAtp3%
 2500 LDAtp3%+1:ADC#0.STAtp3%+1
 2510 SEC:LDAtp3%:SBCmz%
 2520 LDAtp3%+1:SBCmz%+1:BCSmexit
 2530 JMPmbc
 2540 .mexit RTS
 2550 ]:NEXT·ENDPROC
 2560 :
 2570 END
```

Listing 2

```
  10 REM Program DIVlarg
  20 REM Version B.4.2c
  30 REM Author  Barry W Fenson
  40 REM BEEBUG  July 1993
  50 REM Program subject to copyright
```

```
  60 :
 100 INPUT"Maximum no. of digits "m%:ma
x%-m%+2 bytes%-max% 2
 110 PROCinit
 120 PROCassemble
 130 CLS
 140 ?ar%=?opd1p%:ar%?1-opd1p%?1-CALLcl
ear
 150 ?ar%=?opd2p%:ar%?1=opd2p%?1:CALLcl
ear
 160 ?ar%=?resp%:ar%?1=resp%?1:CALLclea
r
 170 ?ar%=?remp% ar%?1=remp%?1:CALLclea
r
 180 ?ar%=?partp%:ar%?1=partp%?1:CALLcl
ear
 190 ?tp1%=opd1%MOD256:tp1%?1=opd1%DIV2
56
 200 PROCenter("Dividend")
 210 ?tp1%=opd2%MOD256:tp1%?1=opd2%DIV2
56
 220 PROCenter("Divisor")
 230 err%=USRdiv
 240 IFerr%>0 PRINT''"Division by zero!
":END
 250 PROCdisplay("Quotient ",resp%)
 260 PROCdisplay("Remainder ",remp%)
 270 END
 280 :
1000 DEF PROCinit
1010 DIMbu% max%,opd1% bytes%,opd2% byt
es%,res% bytes%,rem% bytes%,part% bytes%
1020 DIMmc% &2C0:opd1p%=&70
1030 opd2p%=&72:bup%=&74
1040 tp1%=&76:tp2%-&78
1050 tp3%-&7A:len%-&7C
1060 mz% &7E.x% &80
1070 flag%-&82.resp%-&84
1080 cnt%-&86.ar% &88
1090 rev%-&8A-aux%-&8C
1100 y% &8E cy%-&90
1110 remp%-&92:partp%-&94
1120 oswrch-&FFEE
1130 ?opd1p%=opd1%MOD256:opd1p%?1-opd1%
DIV256
1140 ?opd2p%-opd2%MOD256:opd2p%?1-opd2%
DIV256
1150 ?bup%-bu%MOD256 bup%?1-bu%DIV256
1160 ?resp%-res%MOD256:resp%?1-res%DIV2
56
1170 ?remp%-rem%MOD256:remp%?1=rem%DIV2
56
1180 ?mz%-bytes%MOD256:mz%?1-bytes%DIV2
56
1190 ?partp%=part%MOD256:partp%?1=part%
DIV256
1200 ENDPROC
1210 :
1220 DEF PROCenter(N$)
1230 PRINT'"Enter "N$" as max ";m%;" DE
C digits;"'" MS Digit first & end with R
eturn"
1240 ?ar%=?bup%:ar%?1=bup%?1:CALLclear
1250 I%=0
1260 REPEAT
1270 A1%=GET-48:IFA1%=79I%=I%-1:VDU127:
GOTO1270
1280 IFA1%<10ANDA1%>-1PRINT;A1%;:I%?bu%
=A1%
1290 I%=I%+1
1300 UNTILI%=max%ORA1%=-35
1310 IFI%=1GOTO130
1320 ?len%=(I%-2)MOD256:len%?1=(I%-2)DI
V256
1330 ?bup%=bu%MOD256:bup%?1=bu%DIV256
1340 CALLinpt
1350 ENDPROC
1360 :
1370 DEF PROCdisplay(N$,T%,
1380 IFN$<>""PRINT'N$" = "'
1390 ?ar%=?T%:ar%?1=T%?1
1400 CALLdisp
1410 PRINT'
1420 ENDPROC
1430 :
1440 DEF PROCassemble
1450 FORpass%-0TO2STEP2
1460 P%=mc%:[OPTpass%
1470 .inpt LDY#0
1480 LDAlen%:STAcnt%
1490 LDAlen%+1:STAcnt%+1
1500 CLC:LDAbup%:ADCcnt%:STAbup%
```

```
1510 LDAbup%+1:ADCcnt%+1:STAbup%+1
1520 .iloop LDA(bup%),Y:STA(tp1%),Y
1530 SEC:LDAcnt%:SBC#1:STAcnt%
1540 LDAcnt%+1:SBC#0:STAcnt%+1
1550 BCC1exit
1560 SEC:LDAbup%:SBC#1:STAbup%
1570 LDAbup%+1:SBC#0:STAbup%+1
1580 LDA(bup%),Y
1590 ASLA:ASLA:ASLA:ASLA
1600 ORA(tp1%),Y:STA(tp1%),Y
1610 CLC:LDAtp1%:ADC#1:STAtp1%
1620 LDAtp1%+1:ADC#0:STAtp1%+1
1630 SEC.LDAbup%:SBC#1:STAbup%
1640 LDAbup%+1:SBC#0:STAbup%+1
1650 SBC:LDAcnt%:SBC#1:STAcnt%
1660 LDAcnt%+1:SBC#0:STAcnt%+1
1670 BCSiloop
1680 .iexit RTS
1690 :
1700 .disp LDY#0:LDX#0
1710 LDAmz% STAcnt%:LDAmz%+1:STAcnt%+1
1720 .brd CLC:LDAar%:ADCcnt%:STAtp1%:ST
Atp2%
1730 LDAar%+1:ADCcnt%+1:STAtp1%+1:STAtp
2%+1
1740 LDA(tp1%),Y:LSRA:LSRA:LSRA:LSRA
1750 CPX#1:BEQpnt
1760 CMP#0:BEQnxt1.LDX#1
1770 .pnt CLC:ADC#48:JSRoswrch
1780 .nxt1 LDA(tp2%),Y:AND#&F
1790 CPX#1:BEQpnt2
1800 CMP#0:BEQrpt:LDX#1
1810 .pnt2 CLC.ADC#48:JSRoswrch
1820 .rpt SEC:LDAcnt%:SBC#1:STAcnt%
1830 LDAcnt%+1.SBC#0.STAcnt%+1
1840 BITcnt%+1.BPLbrd
1850 TXA.BNEdexit
1860 LDA#48:JSRoswrch
1870 .dexit RTS
1880 :
1890 .clear LDY#0
1900 LDAmz%:STAcnt%:LDAmz%+1:STAcnt%+1
1910 .brc CLC:LDAar%:ADCcnt%:STAtp2%
1920 LDAar%+1:ADCcnt%+1:STAtp2%+1
1930 LDA#0:STA(tp2%),Y
1940 SEC:LDAcnt%:SBC#1:STAcnt%
1950 LDAcnt%+1":SBC#0:STAcnt%+1
1960 BITcnt%+1:BPLbrc
1970 RTS
1980 :
1990 .div LDY#0
2000 STYaux%
2010 LDAmz%:STAcnt%:LDAmz%+1:STAcnt%+1
2020 ASLcnt%:ROLcnt%+1:INCcnt%:BNEckz
2030 INCcnt%+1
2040 .ckz LDAmz%.STAx%:LDAmz%+1:STAx%+1
2050 STYy%:STYy%+1
2060 .dz CLC:LDAopd2p%:ADCy%:STAtp1%
2070 LDAopd2p%+1:ADCy%+1:STAtp1%+1
2080 LDAcy%.ORA(tp1%),Y:STAcy%
2090 INCy%:BNEdn1:INCy%+1
2100 .dn1 LDAx%:BNEdn2:DECx%+1
2110 .dn2 DECx%:LDAx%:ORAx%+1:BNEdz
2120 LDAcy%:CMP#0:BNEdbc
2130 JMPdverr
2140 .dbc LDAopd1p%+1:STAy%+1:LDAopd1p%
:STAy%
2150 JSRrv
2160 DECcnt%:BNErdb
2170 LDAcnt%+1:BNEnofin
2180 JMPdvexit
2190 .nofin DECcnt%+1
2200 .rdb CLC:LDAremp%+1:STAy%+1:LDArem
p%:STAy%
2210 JSRrv
2220 STYaux%
2230 .sbc STYy%:STYy%+1
2240 LDAmz%:STAx%:LDAmz%+1:STAx%+1
2250 LDA#1:STAcy%
2260 .ind CLC:LDAremp%:ADCy%:STAtp1%
2270 LDAremp%+1:ADCy%+1:STAtp1%+1
2280 CLC:LDAopd2p%:ADCy%:STAtp2%
2290 LDAopd2p%+1:ADCy%+1:STAtp2%+1
2300 CLC:LDApartp%:ADCy%:STAtp3%
2310 LDApartp%+1:ADCy%+1:STAtp3%+1
2320 LDAcy%:BEQcy0.SEC
2330 .cy0 STYcy%
2340 SED
2350 LDA(tp1%),Y:SBC(tp2%),Y:STA(tp3%),
Y
2360 STYcy%:BCCcc1:LDA#1:STAcy%
2370 .cc1 INCy% BNEdn3:INCy%+1
```

```
2380 .dn3 LDAx%:BNEdn4.DECx%+1
2390 .dn4 DECx%
2400 LDAx% ORAx%+1
2410 CLD
2420 BNEind
2430 LDAcy%:CMP#1:BEQnoloop
2440 JMPdbc
2450 .noloop CLC:LDAaux%:ADC#&10:STAaux
%
2460 LDAremp%:STAx%:LDAremp%+1:STAx%+1
2470 LDApartp%:STAremp%:LDApartp%+1:STA
remp%+1
2480 LDAx%:STApartp%:LDAx%+1:STApartp%+
1
2490 JMPsbc
2500 .dvex.t.LDAmz% STAcnt%:LDAmz%+1:ST
Acnt%+1
2510 .dbcd CLC:LDAopd1p%:ADCcnt%:STAtp1
%
2520 LDAopd1p%+1 ADCcnt%+1 STAtp1%+1
2530 CLC:LDAresp%:ADCcnt%:STAtp2%
2540 LDAresp%+1:ADCcnt%+1:STAtp2%+1
2550 LDA(tp1%),Y:STA(tp2%),Y
2560 SEC:LDAcnt%:SBC#1:STAcnt%
2570 LDAcnt%+1:SBC#0:STAcnt%+1
2580 BITcnt%+1:BPLdbcd
2590 RTS
2600 .rv LDAy%+1:STArev%+1:LDAy%:STArev
%
2610 LDAmz%:STAx%:LDAmz%+1:STAx%+1
2620 STYy%:STYy%+1
2630 .shd CLC:LDArev%:ADCy%:STAtp1%
2640 LDArev%+1:ADCy%+1:STAtp1%+1
2650 LDA(tp1%),Y:PHA:AND#&F
2660 ASLaux%.ORAaux%
2670 ROLA:ROLA:ROLA:ROLA
2680 STA(tp1%),Y.PLA:AND#&F0:STAaux%
2690 INCy%:BNEdn5:INCy%+1
2700 .dn5 LDAx%:BNEdn6:DECx%+1
2710 .dn6 DECx%
2720 LDAx% ORAx%+1:BNEshd
2730 RTS
2740 :
2750 .dverr LDA#1
2760 RTS
2770 ]:NEXT ENDPROC
2780 :
2790 END
```

More Hide and Seek (continued from page 40)

```
2240 ENDPROC
2250 :
2260 DEF PROCstartnewgame
2270 PROChowhard
2280 FOR X%=1 TO 6
2290 HAZARD%(X%)=RND(15)+1
2300 Y%=-1:REPEAT:Y%=Y%+1
2310 UNTIL HAZARD%(Y%) HAZARD%(X%) OR Y
%=X%-1
2320 IF HAZARD%(Y%)-HAZARD%(X%) GOTO 22
90
2330 NEXT X%
2340 YOUAREHERE%=HAZARD%(6)
2350 PLAYING%=TRUE
2360 IF DIFF%-1 HAZARD%(1,--1-HAZARD%(3
)-1
2370 ENDPROC
2380 :
2390 DEF PROChowhard
2400 CLS
2410 PRINT"How difficult do you want"'"
the game to be?"''
2420 INPUT"(1,2 OR 3)";DIFF%
2430 IF DIFF%<1 OR DIFF%>3 GOTO 2420
2440 ENDPROC
2450 :
2460 REM ROOM CONNECTIONS
2470 DATA2,4,7,0,0
2480 DATA1,3,4,6,0
2490 DATA2,10,6,0,0
2500 DATA7,1,2,5,8
2510 DATA4,6,9,8,0
2520 DATA5,2,3,10,9
2530 DATA1,4,11,14,0
2540 DATA4,5,12,11,0
2550 DATA5,6,13,12,0
2560 DATA3,16,6,13,0
2570 DATA8,12,15,14,7
2580 DATA11,8,9,13,0
2590 DATA10,16,15,12,9
2600 DATA7,11,15,0,0
2610 DATA14,11,13,16,0
2620 DATA15,13,10,0,0
2630 :
2640 DEF PROCinfo
2650 FOR X=1TO 5
2660 PRINT HAZARD%(X);
2670 NEXT
2680 ENDPROC
```

Mr Toad's Machine Code Corner

Mr Toad thinks it's about time he amswered some questions.

Happy July, frog fanciers. The old column has been bringing in some interesting letters lately, and Mr T has been corresponding with Tim Parsons of Flitwick, Arthur Adams of Upminster and James Sugden of Cleckheaton. The last two gentlemen are well past retirement age; both are deeply into computing and 6502 programming in particular - who says us old 'uns are past it? You certainly can teach an old frog new tricks. However, let's look at a query from Tim Parsons.

In the nicest possible way, Tim was really telling me off - and rightly so - for sprinkling Master-only instructions so carelessly around my listings. I promised him that in future I would be more thoughtful; not everyone has a Master, and all I need to do is run my own 'Master Blaster' ROM over my assembly texts. Tim asked whether putting the 65C02 chip into his old 'B' would enable it to cope with the new instructions.

Well, yes, in a way it would - it would enable the 'B' to run *assembled* code containing the new opcodes, provided that the programs were written entirely *legally*. The trouble is, in BEEBUG you're not dealing with assembled code; programs are normally Basic assembler listings. The Basic 2 ROM of the 'B' has never heard of the new instructions and doesn't know that you've supercharged the old heap, so it will come up with an error message every time it meets a new instruction. You can get round this by assembling the new opcodes 'by hand' - each time the listing says PHX, you type in EQUB &DA; when it says INA you type EQUB &1A

The trouble is, not all instructions are one byte only, some need addresses. STZ, for example: STZ &12AB would become EQUB &9C:EQUB &AB:EQUB &12, because addresses are stored LSB-MSB (or posterior-about-face, to put it in technical terms.) Having made that point, what you'd really write is EQUB &9C:EQUW &12AB, but it's still a pain. Zero-page STZ is opcode &64, not &9C, so STZ &FF would be EQUB &64:EQUB &FF or you could waste a byte and put EQUB &9C:EQUW &FF. STZ &12AB,X would be EQUB &9E:EQUW &12AB and STZ &80,X becomes EQUB &74:EQUB &80

The new addressing mode, 'indirect zero page' creates similar problems and BRA (Mr T's pet hate, as you know) involves calculating the jump displacement. What's worse, it's not usually addresses you're dealing with in assembler listings, but labels. You'd have to ask the Beeb for the address corresponding to each label - PRINT ~toadLoop - and then you'd need to do them all again if, when debugging, you found that a typing error had made them wrong by even a single byte. And, as I say, you'd still have no assurance that the program wouldn't fail even then because of other differences between the 'B' and the Master.

Gentle reader, please note that here we're talking about getting the 'naughty' instructions to run on a MODIFIED BBC B, not about replacing them with other instructions which exist in the old set, so as to make the code run on a normal 'B', which is what we were looking at a few months back.

Neither strategy is really a lot of fun; you might just as well buy a Master and get all the other goodies as well; View 3.0, ViewSheet, standard Sideways RAM and so on. This is essentially what I told Tim, only to find that he'd already come to the same conclusion himself and bought a second-hand Master from Beebug.

In fact, he got a Turbo. Lucky man because, although having the co-pro running all the time can create some annoying problems with certain pieces of software, it gives you a lot more memory with View, and - as long as you load it in from disc each time - with Hibasic. The extra speed is also very useful with ViewSheet and, often, with Basic, although some Basic programs have bits which run TOO fast on the co-processor (Basic programmers - please think of this and use TIME rather than FOR NEXT when writing delay loops). Anyway, I digress. (As usual. Ed.) Tim was now wondering why you automatically get extra memory with View if the co-processor is active, but not with ViewSheet, Basic etc. Being one of the idlest life forms in the Galaxy, I'll just repeat my letter to Tim, even if Mr Williams does knock 10% off my honourarium. (Worse - I'll knock off 50 pence! Ed.)

Firstly, let's take a good look at what the so-called 'Tube' really is. On the Turbo board you've got a faster processor, the 65C12, plus 64K of RAM. It's virtually a second computer, in the sense that code running on the new chip, in the new RAM, is entirely independent of the 'old' hardware - at least until it wants input from the keyboard or needs to output to the screen. The existing setup looks after the keyboard, filing, the screen and so on, which gives another advantage speed-wise, since the running of the code isn't constantly halted while the processor deals with interrupts to handle these things. The Turbo, therefore, is more than twice as fast at running machine code

The word 'Tube' is often wrongly used to refer to the Turbo board, but actually it's the chip which transfers data between the two boards: an *asynchronous* link, meaning that the Turbo board is not only running at twice the speed of the main board, but that the correspondence may not be perfectly exact - at any given moment the two boards may not be at precisely the same state of their clock cycles. What's more, the Turbo is only one of a range of co-processors that might be fitted; there's the 512, of course, or the Z80. These non-6502 boards run their own machine-code, so the Tube is designed just to transfer single bytes, regardless of their meaning and of any disparity in the clock-rates. The Tube's data and status registers are mapped in the host (i.e. main computer) memory map from &FEE0 to &FEE7 - in our old friend SHEILA - and you can use various OSBYTES to read and write bytes across the Tube, or else write your own 'MOS' code to deal directly with the Tube registers, which involves some tricky timing, working literally with millionths of a second. Mr T used the latter method for the memory-editor in the Toad ROM 90, purely for the fun of it. I won't even attempt to give details here; you need a manual such as Watford's 'Advanced Reference Manual for the BBC Master' or David Atherton's excellent 'Master Operating System' (Dabs Press). Beebug stocks both

The Turbo board gives not only more speed, but much more RAM: since most of the MOS is on the host board, the space from &8000 on up is full of RAM, most of it free for the user. There's also a gain of a few pages at the bottom end of RAM, since the Turbo board needs less workspace there.

What happens when a Turbo *CONFIGUREd TUBE starts up with, say, Basic or ViewSheet? The MOS copies the Basic or ViewSheet ROM across the Tube, one byte at a time (but it's very fast!), and it starts running. You gain a bit of extra memory at the low end of RAM; under Basic, for example, PAGE is at &800 rather than &E00, but you gain no memory at the top end. Basic, etc. are assembled to run between &8000 and &BFFF; if you wanted them to run elsewhere, you'd need a version in which all the JMPs and JSRs which refer to

addresses inside the ROM were recalculated to take account of the ROM now being at a new address. If you had the assembly text of Basic, you'd just alter P% to the new start - but you don't have the assembly text So, ROMs like Basic, when copied across the Tube to the Turbo board, must be copied to &8000 on that board They take up 16k of RAM there, but, as we have seen, above that 16k there is a lovely, juicy stretch of almost another 16k of free RAM, in the area where the MOS is on the host board... and you can't use it. Basic, View, etc are set up only to use RAM *below* &8000, they can't work with the RAM *above* themselves, from &C000 on up

What we need, then, is a version of our ROM which not only gets copied across the Tube, but gets copied to a much higher address, leaving all the free RAM below it - but that version must have its internal JMP and JSR addresses altered automatically. Only View is made to do this: it is copied faithfully to &8000 on the Turbo board, then most of it gets recopied up higher with the addresses recalculated, and the 'ghost' copy at &8000 is redundant and can be overwritten. Thus View offers 48,326 free bytes instead of 28,926. To do the same with Basic, you need to load in HIBASIC from the disc which comes with the Turbo board; this is just Basic assembled with P% set to &B800, though the MOS also has to be made to call the new address. Under the new MOS chip this is automatic with Basic as well as View

That's what I wrote to Tim Parsons, more or less, and he was kind enough to state that for once I didn't leave him more confused than he was before. I wonder... That's it for this month, mammal types Don't forget that Beebug members traditionally mungle their weebles on the 27th; Mr T will be asking how you got on in the next issue, as well as publishing a listing which anti-aliases the contours of your local Bishop's gaiters. Until then, gentle reader, I bid thee farewell.

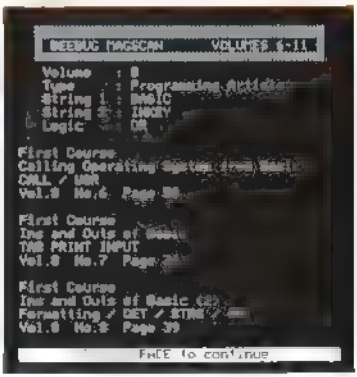
Magscan Comprehensive Magazine Database for the BBC Micro and the Master [illegible]

***An updated version of Magscan, which contains the complete indexes to all BEEBUG magazines from* Volume 1 Issue 1 *to* Volume 11**

Magscan allows you to locate instantly all references to any chosen subject mentioned anywhere in the 110 issues of BEEBUG magazine to date

Just type in one or two descriptive words (using AND/OR logic), and you can find any article or program you need, together with a brief description and reference to the volume issue and page numbers You can also perform a search by article type and/or volume number

The Magscan database can be easily updated to include future magazines. Annual updates are available from BEEBUG for existing Magscan users



Some of the features Magscan offers include:

- [illegible] access to all BEEBUG magazines
- rapid keyword search
- flexible search by volume number article type and up to two keywords
- keyword entry with selectable AND/OR logic
- extensive on-screen help
- hard copy option
- easily updatable to include future magazines
- yearly updates available from BEEBUG

Magscan with disc and manual **£9.95+p&p**

Stock codes	0005a	5.25"disc 40 track DFS
	0006a	5.25"disc 80 track DFS
	[illegible]	3.5" ADFS disc

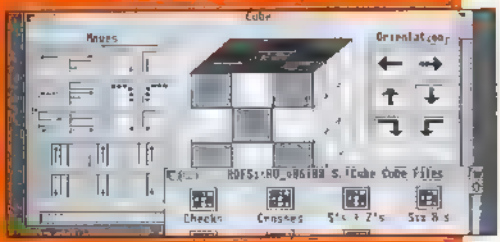
Magscan update **£4.75+p&p**

Stock codes	001[illegible]a	5.25"disc 40 track DFS
	0010a	[illegible] disc 80 track DFS
	[illegible]	3.5" ADFS disc

RISC Developments Ltd, 117 Hatfield Road, St Albans, Herts AL1 4JS. Tel 0727 840303 Fax 0727 860263

RISC User

The number one subscription magazine for the Archimedes



RISC User, probably the most popular subscription magazine for the Archimedes, offers all the information you need as an Archimedes user. In every issue of RISC User you will find a wealth of articles and programs with professionally written reviews, lively news, help and advice for beginners and experienced users, and items of home entertainment.
The B5 size of RISC User allows a sophisticated design, big colour illustrations and pages full of information, and yet is still a convenient size to assemble into an easy-to-use reference library. Altogether, in its six years of existence, RISC User has established a reputation for a professional magazine with accurate, objective and informed articles of real practical use to all users of Acorn's range of RISC computers.

GROUP SURVEY - WORD PROCESSORS
A critical look at the major players in the word processor market.

JUST A PAGE: PROGRAMS TO TYPE AND USE
A new column offering this month a Mode Changer, Tiny Catalogues and CMOS RAM Procedures.

SHERSTON'S BADGER TRAILS
A look at the award winning education package - a multi-media study of badgers, accompanied by a short video.

THREE IMAGE CD'S: PHOTO-QUALITY CLIP ART
A comparison review of three photo-libraries on CD discs.

WIMP TOPICS
A major series aimed at readers interested in Wimp programs and Wimp programming. Each article looks at aspects of a particular topic.

WRITE-BACK
The readers' section of RISC User for comment, help, information - a magazine version of a bulletin board.

INTO THE ARC
A regular series for beginners.

TECHNICAL QUERIES
A column which answers your technical queries.

THE DOS SURVIVAL GUIDE
A series of articles on how to use the PC Emulator.

PD SOFTWARE COLUMN: SELECT AND COLLECT
A look at the best of Public Domain software.

SOLVE THE CUBE
A computer version of the famous Rubic's cube.

SUBSCRIPTION DETAILS

As a member of BEEBUG you may subscribe to RISC User for the reduced rate of £18.40 (a saving of £1.50 on the normal subscription rate). Overseas subscription rates are as follows: £27.50 Europe and Eire, £33.50 Middle East, £36.50 Americas & Africa, £39.50 Elsewhere

Please keep sending in any tips for all BBC and Master computers. Remember, if your hint gets published, there's money in it!

Changing Colours in Mode 7

David Robinson

For those of you who are mode 7 addicts, but are frustrated by its inability to change the overall display foreground and background colours, here's a quick fix

Normally, when changing colour or creating other effects, the display control characters that are 'printed' to the screen only affect one line of text

The following Basic program overcomes this limitation by setting up foreground and background colours that are retained when the screen is scrolled

```
10 red$=CHR$ 12+
2[illegible]
3[illegible]
4[illegible]
5[illegible]
6[illegible]
70[illegible]
80[illegible]
90[illegible]
[illegible]
[illegible]
[illegible]
MODE
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
END
```

The program works by printing colour control characters down the left-hand edge of the screen and then defining a text window which avoids these positions to prevent them being overwritten. Line 100 sets the foreground and background colours, and currently defines a blue background with cyan text. To change this, substitute any of the colours defined in lines 10 to 70 for blu$ and cya$. Line 80 sets bkg$ to the 'new background' code, not a colour.

Testing Osbyte Routines

Jonathan Temple

As many OSBYTE routines are not implemented as *FX commands, it is not always easy to test them from Basic and see their results. When executed, the following function key definition will prompt you to enter the values for A (the OSBYTE call number), and the X and Y registers. It will then call the specified OSBYTE routine and display the results.

```
*KEY[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
```

Wordwise Plus and VDU Codes

Wordwise Plus and BBC Basic share many common features, including the Basic statement VDU. The VDU codes can be strung together in the normal way and used to print out characters on the screen that could not otherwise be produced. Try the following segment program:

```
CLS
[illegible]
[illegible]
PRINT "Double height in colour."
[illegible]
IF GET THEN END
```

This will clear the screen and print a double height banner in yellow on a red background.

VDU 31 can be used to good effect in a similar manner to position text on the screen. VDU 31 is equivalent to the Basic TAB(x,y). For example:

```
CLS
[illegible]
REPEAT
[illegible]
PRINT "Hello"
[illegible]
[illegible]
UNTIL [illegible]
IF GET THEN END
```

However, be very careful not to execute a VDU 22 (select screen mode) command from Wordwise Plus. This can cause all your text and programs to be lost. B

Personal Ads

BEEBUG members may advertise unwanted computer hardware and software through personal ads (including 'wants' in BEEBUG. These are completely free of charge but please keep your ad as short as possible. Although we will try to include all ads received, we reserve the right to edit or reject any if necessary. Any ads which cannot be accommodated in one issue will be held over to the next, so please advise us if you do not wish us to do this. We will accept adverts for software, but prospective purchasers should ensure that they always receive original copies including documentation to avoid any abuse of this facility.

We also accept members' Business Ads at the rate of 40p per word (inclusive of VAT) and these will be featured separately. Please send all ads, personal and business, to MEMBERS' ADS, BEEBUG, 117 Hatfield Road, St. Albans, Herts AL1 4JS.

WANTED: Vigilen PC style case for M128. Tel. (0727) 830264

WANTED: The Advanced Basic ROM User Guide for the BBC Micro (by Colin Pharo). Tel 071 240 2363 daytime or (0932) 225383 eves.

WANTED: Instructions on the Replay ROM. Tel (0934) 622209

Numerous items of BBC B software, Hardware and Firmware for sale. Tel. 051-228 7136 after 6pm.

WANTED: Turbo internal 6502 2nd processor for Master Tel. 081-894-0891

Master Smart cartridge £15. Overview suite £40, Cortex Master Bank Manager with business utilities £20, Gemini Money Management (BBC) £5, Master Mini-Office II £10. Play It Again Sam13, Repton 2, Citadel, Barbarian, Exchange Location, Death Star, BEEBUG Filer Editor, Basic Booster, ViewChart all £8 each, Spellmaster ROM £30, View Datband Guide c w disc £10, Miracle Modem and Command ROM £70, Master Turbo and Alternative MOS £130, 20 unformatted DSDD reversable discs c/w labels, tabs £10, Sharp MZR/k to centronics interface £10. Tel. ((0268) 448

BBC B 2x Watford DSDD drives all boxed. ROM expansion board, Stands ROMs etc. as new condition fully working order £150 Tel. (0245) 450056 home

Master 512 also Turbo SP card, Cumana combination 5.25/3.5 floppy drives, tape (till View, Overview package, Wordwise Plus, Interbase DTP, Art, Graphics, PC Software Canon PW1080 printer, will split, Morley teletext adaptor, Pace modems, library of manuals, books, magazines, call for list, full set of BEEBUG mags Vol.3 No.1 to date Tel (0932) 865221

BBC B issue 7, DFS twin 40/80 disc drives, colour monitor with stand, joystick, sideways and shadow RAM boards (Watford), wide range of ROMs for word processing, communications and utilities, extensive collection of software, plus manuals, mostly disc based, but some tapes included plus data recorder, a large collection of BEEBUG and Disc User magazines is part of the package. Buyer to arrange collection £250 Tel (0732) 359954 Would prefer to sell as a complete package

BEEBUG Magic Modem. BEEBUG Command ROM, Comms software £45 o.n.o. Tel. (0454) 2xbbx8 after 6pm.

Master Compact c/w Acorn mono monitor, needs new EPROM hence £70, Master Reference Manuals 1&2 £10, Hyperdriver ROM £8. Tel (0742) 361307 eves

WANTED: Apex ROM/RAM board for BBC micro, made by Altair Electronics, top price paid, circuit diagram alone would help, damaged board considered. Tel. 071-727 6752.

WANTED: Solidisc 4Mbz RAM board for BBC B with software and manuals, also interface and adaptor board to connect hard drive (Seagate) Tel (0773) 829543

WANTED: Software for Torch Z80 addon processor for BBC B particularly Perfect Writer, Perfect Speller Tel (0727) 830264

WANTED: Acorn Teletext adaptor (must be old type 1984-1986) and Acorn Bridge (also old type 1984-1991) £75 offered for each Tel (0371) 820527 or 820176

Structural Design Programs

For engineers and educational establishments to solve practical design problems on the BBC B & Master

Titles include hydraulic RAM, bolt group, polynomial, section properties, vector polygon, catenary, cable combinations and redundant frame analysis.

Programs may be run interactively or with input data edited under view. Comprehensive system also for the Z88.

For details, send s.a.e. to: Frameworks (SDCS), 58 Charlotte Road, London EC2A 3QT

Archimedes 420/1, 40Mb IDE drive 4Mb RAM, colour monitor, over £1,000 software, magazines all original package, Star LC24-200 printer £1,200 o.n.o. Tel. 081 551 1839 anytime.

BBC powerusers: BBC B issue 7 fitted with 1770 disc controller. DFS & ADFS and ATPL Sideways ROM board featuring Basic, Forth, Pascal, Logo, View, Viewstore, Connestar Disc Doctor and Printmaster ROMs plus 16Kb RAM. Microvitec Cub monitor and stand, dual DS/80T disc drives, Viglen 20Mb Winchester drive, Acorn Data Recorder, Pace Nightingale modem, AMX mouse + 5 software packages, AMS 300 Music Synthesiser, joysticks, 6502 second processor + 1 CSD pc system, Z80 second processor and all original software. ACS2016 Cambridge co-processor upgraded to 10MHz + all original software. Torch Graduate PC system fitted with extra 256Kb RAM, Brother HR15 Daisywheel printer, all cables plus additional software everything in excellent condition, over 90 books and manuals, list available, and copies of AU, ML and BEEBUG issues since 1983, sensible offers for individual items, or bargains for bundles. Tel. 081-668 7167

WANTED: Watford co-pro adaptor or Acorn Universal second processor unit, as well as Acornsoft's Strategy game Go! (either on disc or tape). Write to, Miroslaw Bobrowski, Waszyntona 22 A m.41. 15-274 Bialystok, POLAND

Master 512 User Guide and Master 512 Technical Guide (£15 for both) Write to Miroslaw Bobrowski, Waszyntona 22-A m.41, 15-274 Bialystok, POLAND.

Acorn A3000 2Mb RAM, RISC OS 3, Acorn colour monitor with stand and software, Learning Curve £475, Watford hand scanner (mono), Impression £100, Revelation II £40, Artisan £30, Programmer's Reference Manuals OS2, £25 BBC Basic Guide £10, Archimedes Operating System £7 Archimedes Assembley Language £7 Wimp Programming for A £6 £70 ono sold as pair. Will sell separately Tel (0859) 571042

WANTED: New to the 'B' No it's not dead! BEEBUG mags, disc based educational software, games, at reasonable prices. Before it's all gone anything accepted rather than thrown away 'cos forever! I would like to control Lego 9 volt models, any interface instructions please, also any 'Red Box' gear Tel (0904) 622209

Music software; Music Master with microphone interface 5.25" disc, handbook, Mupadra Recorder Tutor with Ensemble, Duet and Classroom network packs (5x5.25" discs, handbooks and cassettes), Micro maestro with 5.25" disc and 6 cassettes, all for £28 including postage (worth £179). Tel. (0256) 27018

Can anyone lend or sell me a copy of Interbase and/or Interchart manuals? Also help with Spellcheck would be appreciated. Tel. ((0738) 812186

BBC B with dual 40/80T disc drive, Centronic GLP printer colour monitor, manuals £50. Tel 061-969 9653

Little used BBC B, ROM Box, Wordwise, other ROMs and programs £90. Unused Z80 processor £70, double 40/80T disc drive unused £30 Epson FX long carriage printer £70 offers? Tel 051 489 6346.

FMS Publisher ROM with 5 discs, includes 36 fonts £20 (cost £50), Printwise 40/80T disc £10, Pixel Perfect DTP 4 discs £10, Studio 8 40/80T disc £5, Structured Basic (National Extension College) £5, Creative Graphics £5, Complete Mouse User Guide inc. disc £5. Programmers Trouble Shooting Guide £3, all prices inc. postage. Tel. (0252) 710219

WANTED: DFS 1770 chip for BBC B, purchase or loan of Welcome Guide or BBC Master Compact and similarly for View and Viewsheet guides. Tel (0642) 710345

BEEBUG's FUTURE

The main reason for this letter is to express my profound sorrow that BEEBUG's days appear to be numbered. I am sure that all your other subscribers must be equally horrified. All hobbyists need a forum, and surely 8-bit users must regard their vice as a hobby. I enjoy Meccano modelling - a youthful ambition at last realised - but I need meetings and exhibitions to show off my work and learn from my peers. And so it is with our 8-bit Beebs. You are our last forum. Acorn User gave up on us some time ago. If you give us up, where can we turn?

So what is the problem? Is it circulation, material, or money? I suppose they are all inter-related. I can understand that circulation will fall as more people upgrade to 32-bit machines, but according to Acorn User six months ago, there are still hundreds of thousands of Beebs and masters in use. There must be thousands of users out there who would gladly subscribe if they knew of your existence. Would more publicity help?

Is it material? You are still running some fascinating series and lone articles, and have reproduced some excellent articles and games from years gone by. Personally, I would be more than happy if you were to start at Vol.1 and work right through the last ten years all over again. I might find time to read it all, and what a treasury I would learn!

I expect it all comes down to money. Well there are several things you could do. You are cutting the size down by several pages. Cut it down some more. Or cut the issues down to five or six a year. Or cut down the actual physical quality of BB. In the early days the paper was poor, the printing was poor, and there was no colour. Go back to cheaper production. I don't know how much you pay regular contributors, but do they really want payment? I would find the mere fact of publication sufficient reward. I can think of one contributor of excellent stimulating articles, whom I now write to personally, who I am sure gets more satisfaction from the publicity than from the cheque I presume you send him.

You would have to keep on charging the same subscription of course. Why don't you sound out subscribers and find out what they would stand for? I would gladly go on paying the same for a reduced service in order to keep the forum alive. Other suggestions are to charge more for the same service, issue a small monthly news-sheet, or give us a few pages in RISC User, which we would subscribe to instead.

BEEBUG is a terrific publication, and we don't want to lose it. We need that forum.

Arthur Adams

We have devoted all of this month's Postbag to Mr. Adams letter. It is by no means the only letter which we have received on this subject, but perhaps best encapsulates many of the comments made by other readers. We would also like to encourage further comment on this issue, and on possible suggestions for the future.

In the event that we do cease publication of BEEBUG, one possibility might be for a group of active and enthusiastic readers to continue as a User Group, publishing some form of magazine or newsletter to take over from BEEBUG. We would provide initial help and encouragement, and hopefully maintain an excellent relationship between this new group and ourselves.

BEEBUG MEMBERSHIP

Send applications for membership renewals, membership queries and orders for back issues to the address below. All membership fees, including overseas, should be in pounds sterling drawn (for cheques) on a UK bank. Members may also subscribe to RISC User at a special reduced rate.

RENEWAL RATES FOR BEEBUG MAGAZINE AND MAGAZINE DISC SUBSCRIPTIONS

See May 1993 Editorial for further explanation

The table below shows the renewal rate applying after the June issue 1993 according to the first issue of the renewal period. For joint BEEBUG/RISC User subscriptions add half the appropriate BEEBUG renewal rate to the full RISC User renewal rate; (UK £18.40, Europe & Eire £27.50, Middle East £33.50, Americas & Africa £36.50, Elsewhere £39.50).

Renewal Issue	Issues to go	Mag UK	Mag Europe	Mag Mid-E	Mag Am+Af	Mag Else	Disc UK	Disc O'Seas
Jun	9	16.56	24.75	30.15	32.85	35.55	45.00	50.40
Jul	8	14.72	22.00	26.80	29.20	31.60	40.00	44.80
A/S	7	12.88	19.25	23.45	25.55	27.65	35.00	39.20
Oct	6	11.04	16.50	20.10	21.90	23.70	30.00	33.60
Nov	5	9.20	13.75	16.75	18.25	19.75	25.00	28.00
Dec	4	7.36	11.00	13.40	14.60	15.80	20.00	22.40
J/F '94	3	5.52	8.25	10.05	10.95	11.85	15.00	16.80
Mar '94	2	3.68	5.50	6.70	7.30	7.90	10.00	11.20
Apr '94	1	1.84	2.75	3.35	3.65	3.95	5.00	5.60

BACK ISSUE PRICES (per issue)

All overseas items are sent airmail. We will accept official UK orders for subscriptions and back issues, but please note that there will be a £1 handling charge for orders under £10 which require an invoice. There is no VAT on magazines.

Volume	Magazine	5"Disc	3.5"Disc
6	£1.00	£3.00	£3.00
7	£1.10	£3.50	£3.50
8	£1.30	£4.00	£4.00
9	£1.60	£4.00	£4.00
10	£1.60	£4.75	£4.75
11	£1.90	£4.75	£4.75

POST AND PACKING

Magazines and discs are postcode a
Please add the cost of p&p when ordering.
When ordering several items use the highest price code, plus half the price of each subsequent code. UK maximum £8.

Post Code	UK, BFPO Ch.I	Europe, Eire	Americas,Africa, Mid East	Elsewhere
a	£1.00	£1.60	£2.40	£2.60
b	£2.00	£3.00	£5.00	£5.50

BEEBUG
117 Hatfield Road, St.Albans, Herts AL1 4JS
Tel. St.Albans (0727) 840303, FAX: (0727) 860263
Office hours: 9am-5pm Mon-Fri Showroom hours: 9am-5pm Monday to Saturday
(24hr Answerphone for Connect/Access/Visa orders and subscriptions)

BEEBUG MAGAZINE is produced by RISC Developments Ltd.

Editor: Mike Williams
Assistant Editor: Kristina Lucas
Editorial Assistance: Marshal Anderson
Production Assistant: Sheila Stoneman
Advertising: Sarah Shrive
Subscriptions: Helen O'Sullivan
Managing Editor: Sheridan Williams

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher, RISC Developments Limited.

CONTRIBUTING TO BEEBUG PROGRAMS AND ARTICLES

We are always seeking good quality articles and programs for publication in BEEBUG. All contributions used are paid for at up to £50 per page, but please give us warning of anything substantial that you intend to write. A leaflet 'Notes to Contributors' is available on receipt of an A5 (or larger) SAE.

Please submit your contributions on disc in machine readable form using plain text format if possible for text, but please ensure an adequate written description is also included of your submission and the contents/format of your disc.

In all communication, please quote your membership number.

RISC Developments Ltd (c) 1993

Printed by Arlon Printers (0923) 268328 ISSN - 0263 - 7561

Magazine Disc

July 1993

CENSUS - This month's programs, in the first of a three part series, provide the means to design a questionnaire and to enter data which has been collected ready for subsequent analysis.

WORD SQUARE SOLVER - Following previous programs for creating Word Search puzzles, this program will help you to solve Word Searches whatever the source.

ENHANCING BASIC'S LISTO COMMAND - This program provides useful extensions to the LISTO command for Basic programmers, allowing listings to be presented in more structured and readable formats.

A POOL OF PERMS - Ring the changes with this program to generate permutations as required.

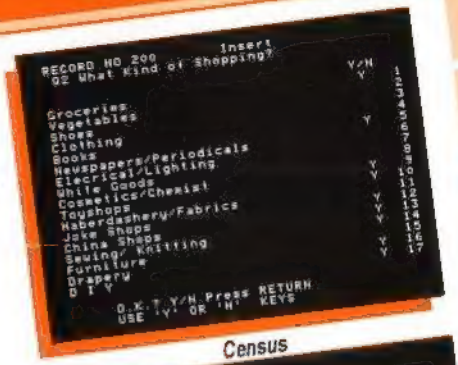
GRAVITY & ORBITS - This last program in the series allows the user to explore gravity fields around a number of bodies up to a maximum of seven.

BEEBUG WORKSHOP - A demonstration of the fast circle drawing technique described in this month's Workshop.

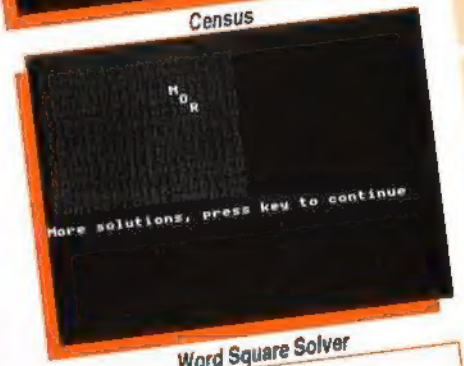
MORE HIDE AND SEEK - The Gribbit program provides both a challenging and enjoyable game, and serves as a demonstration of the art of writing games programs.

VERY BIG NUMBERS (PART 2) - This month we give two programs implementing routines for the multiplication and division of large numbers, incorporated into complete working demos.

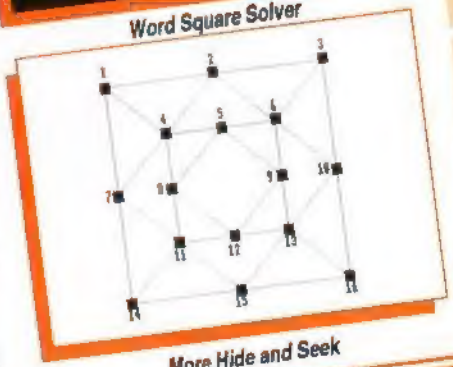
Bibliography for this issue of BEEBUG (Vol.12 No.3).



Census



Word Square Solver



More Hide and Seek

ALL THIS FOR £4.75 (5.25" & 3.5" DISC) + £1 P&P (50p FOR EACH ADDITIONAL ITEM)
Back issues (5.25" and 3.5" discs from Vol.6 No.1) available at the same prices.

FOR DISC (5.25" or 3.5") RENEWAL SUBSCRIPTION RATES (UK AND OVERSEAS) PLEASE SEE TABLE ON FACING PAGE

Prices are inclusive of VAT and postage as applicable. Sterling only please.

Special BEEBUG Members Offer

SPECIAL COMPUTER CONVERSION PACKAGE

As you may be aware Acorn has announced the demise of the BBC Master 128, additionally we have mentioned that BEEBUG magazine is unlikely to continue beyond its twelfth year (April 1994). As such it will become increasingly less cost effective to maintain Acorn's 8-bit computers, and the availability of new software and hardware will almost certainly be reduced to virtually zero. In recognition of this problem BEEBUG are urging you seriously to consider upgrading to one of Acorn's latest RISC computers. For many people this can be a daunting prospect. To help with converting systems we are making various offers aimed at simplifying the transfer of software from your BBC. Our Special Offer falls into five parts.

FANTASTIC OFFERS ON NEW ACORN COMPUTERS

☛ A3010

Code 0172g ***RRP*** £424.68
Offer Price £378 **SAVE £46**

Features: ARM 250 processor, mouse, stereo sound, 2Mb (unformatted) floppy drive, 1Mb RAM (upgradable to 4Mb RAM), mini-expansion slot, serial and parallel interfaces, TV modulator and The Family Solution software pack. Comes complete with EasiWord word processor and Quest for Gold game.

☛ A3010 Learning Curve

Code 0173g ***RRP*** £680
Offer Price £606 **SAVE £74**

Features: In addition to the items in the Family Solution package, the Learning Curve includes 2Mb RAM, colour monitor, PC Emulator and the Genesis Collection (useful in education).

☛ A4000 Home Office

Code 0209g ***RRP*** £999
Offer Price £890 **SAVE £109**

Features: Separate PC style keyboard, ARM 250 processor, 2Mb RAM (upgradable to 4Mb RAM), 80Mb hard drive, colour monitor, 2Mb unformatted floppy drive, serial and parallel interfaces and one mini-expansion slot and network interface slot. Software includes Easiwriter wordprocessor and Desktop Database.

☛ A5000 4Mb HD162

Code 0206g ***RRP*** £1599
Offer Price £1409 **SAVE £190**

Features: ARM 3 processor, separate PC style keyboard, mouse, 4 slot backplane for expansion cards, 2Mb unformatted floppy drive, 4Mb RAM, 162Mb hard drive and multi-scan monitor.

☛ A5000 Learning Curve

Code 0360g ***RRP*** £1445.96
Offer Price £1275 **SAVE £170**

As above but with only 2Mb RAM and 80Mb hard drive, together with a great collection of software including PC Emulator, 1st Word Plus wordprocessor, Acorn DTP, Genesis Plus and Pacmania Game.

These various conversion offers are designed to give you excellent value and a reasonably trouble free change to your new faster computer system. The new system will open up new horizons and give you a greater choice of an ever growing number of software packages.

All these offers are backed up by our telesales, technical support and computer repair services. If you require more information about any of the equipment you may need to transfer your system then reference to BEEBUG magazine issues Volume 8 numbers 9 and 10 will help or telephone us directly for technical advice.

SERIAL LINK SOFTWARE TO EASE TRANSFER OF YOUR FILES

So as to cause you as little upheaval as possible when changing your system we are offering you Ivoryash's Serial Link and lead for only £25.

FREE RISC USER SUBSCRIPTION OFFER

(when you purchase one of the above computers)

We will supply you with a subscription to RISC User magazine to run concurrently with your copies of BEEBUG until April 1994, whether or not your subscription ends before that date.

SOFTWARE OFFER

So that you may continue to use the software to which you are largely used, we are making a special offer on the following software packages to run under the BBC emulator on your new Acorn computer - Wordwise £20, Interword £20, View £32, Masterfile II £5, Intersheet £25.

TRADE IN YOUR EXISTING EQUIPMENT FOR A NEW SYSTEM

You may prefer to keep your BBC and sell it privately (don't forget that members may place free ads in BEEBUG magazine). Should you wish us to take it in part exchange we are currently able to offer the following prices:

BBC Issue 7	**£30**	**BBC Issue 7 with DFS**	**£70**
Master 128	**£100**	**Microvitec monitor**	**£40**

Trade in offers can be applied up to two months after purchase of your new equipment. Trade in prices INCLUDE VAT, all others are ex. VAT.

BEEBUG 117 Hatfield Road, St Albans, Herts AL1 4JS
Tel: 0727 840303 (24 hours) Fax: 0727 860263 All prices are ex VAT and p&p